

**YAŞAR UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES**

(MASTER THESIS)

**VISUALIZATION OF GML DATA WITH XSLT FOR
WEB BASED GIS**

Buket AKILLI

Thesis Advisor: Assoc. Prof. Dr. Murat KOMESLİ

Department of Computer Engineering

Presentation Date: 01.09.2014

Bornova-İZMİR
2014

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of master of science.

Assoc. Prof. Dr. Murat KOMESLİ (Supervisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of master of science.

Prof. Dr. Mehmet Cudi Okur

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of master of science.

Prof. Dr. Vahap TECİM

Prof. Dr. Behzat GÜRKAN
Director of the Graduate School

ABSTRACT

VISUALIZATION OF GML DATA WITH XSLT FOR WEB BASED GIS

AKILLI, Buket

MSc in Computer Engineering

Supervisor: Assoc. Prof. Dr. Murat KOMESLI

September 2014, 94 pages

In the GIS Industry, Development of Web-based Geographic Information Systems (GIS) is performed by popularity of Internet and Web Technologies. The internet has many advantages like working directly on the browser, platform independence and being freely available. There are a number of web-based GIS systems in existence. Each of these systems is developed by using various programming techniques, software and web technologies. So, there are various kinds of visualization results like static images or dynamic complex images with GIS functionality. Users easily obtain maps with using web-based Geographic Information Systems (GIS). On the contrary, the common method of GIS lacks interoperability and efficiency. To overcome these issues, we can apply the most common geographical data format known as Geographic markup language (GML). Geo data can be structured and visualized with this geographical data format. Because of the fact that the abovementioned GML format is a web-based technology, visualization has been made by a specific web-based graphical visualization technique, Scalable Vector Graphics (SVG). Due to the fact that there are more common properties between the two technologies, GML and SVG like being Extensible Markup Language (XML) based and other many features are described in the paper, GML documents have been indispensable appearing in SVG format. Web Feature Service (WFS) is a required technology to obtain a Geographic Markup Language (GML) data for advanced Web-based Geographic Information Systems (GIS). Besides these technologies, Extensible Style Sheet Transformation Language (XSLT) transformation specification also has an important role in the development of efficient web-based Geographic Information Systems. To accomplish the transformation of GML data into Scalable Vector Graphics (SVG), XSLT works together with an XSLT processor. Geo Data can be visualized differently by using different XSLT style sheets. Geo data in the form of GML is first obtained from

available WFS in real time and is then transformed into SVG by XSLT. At this point, it has been realised that Geographical data is needed to style effectively with Cascading Style Sheet (CSS) for supporting layer's visualization rules. As a result of this necessity, the importance of styling with CSS has been understood. In this way, geo data will be visualized as high quality vector maps in browser software and we achieve data interoperability and efficient dissemination of maps. The software built on MapXtreme development kit using the C# programming language in the .Net environment with interactive user interface. The aim of this thesis is to provide an overview of the visualization of Geographic Markup Language (GML) markup data format which is obtained from Web Feature Service (WFS). Construction of a WFS server and client together in the unique web project will also be presented to aid in the presentation of this paper.

Keywords: Visualization, XSLT, GML, SVG, CSS

ÖZET

WEB TABANLI CBS İÇİN XSLT YARDIMIYLA GML VERİLERİN GÖRÜNTÜLENMESİ

AKILLI, Buket

Yüksek Lisans Tezi, Bilgisayar Mühendisliği Bölümü

Tez Danışmanı: Doç. Dr. Murat KOMESLİ

Eylül 2014, 94 sayfa

İnternet ve web teknolojilerinin gelişmesi ile birlikte, Coğrafi Bilgi Sistemleri(CBS) endüstrisinde web tabanlı Coğrafi Bilgi Sistemleri (web CBS) uygulamaları ön plana çıkmıştır. İnternet'in doğrudan web tarayıcı üzerinden çalışmak, platform bağımsız olmak, ucuz olarak elde edilebilmek gibi avantajları bulunmaktadır. Günümüzde birçok web tabanlı olarak çalışan Coğrafi Bilgi Sistemi(CBS) uygulamaları bulunmaktadır. Bütün bu sistemler, çeşitli programlama teknikleri, yazılımlar ve web teknolojileri kullanılarak geliştirilmişlerdir. Bu çeşitliliğin sonucu olarak görüntülenen sonuç haritalar da statik özelliklere sahip veya CBS fonksiyonlarına sahip dinamik özellikteki haritalar olabilmektedir. Kullanıcılar, web CBS kullanarak haritalara web üzerinden kolayca ulaşabilmektedirler. Buna karşın Coğrafi Bilgi sistemlerinde yaygın olarak kullanılan metot olan masaüstü CBS uygulamalarında, uygulamaların birlikte çalışabilmek ve efektif olmaktan yoksun olmak gibi problemleri mevcuttur. Bu gibi problemlerin üstesinden gelmek için Coğrafi Bilgi Sistemleri endüstrisinde en çok tercih edilen veri formatı olan Coğrafi İşaretleme Dili (Geographic Markup Language-GML) formatı çözüm olarak tercih edilmektedir. GML coğrafi veri formatı ile birlikte coğrafi verilere bir yapı kazandırılabilen ve görüntüleme işlemi sağlanabilmektedir. GML, web tabanlı bir teknoloji olduğundan görüntüleme işlemi özel web tabanlı grafik görüntüleme teknolojilerinden Scalable Vector Graphics (SVG) teknolojisi ile sağlanabilmektedir. Bu iki teknoloji arasında bulunan yapısal benzerliklerden başta iki teknolojinin de Extensible Markup Language (XML) tabanlı olması olmak üzere, diğer birçok benzer özelliklerden dolayı GML dokümanları, SVG grafik formatı olarak görüntülenebilmektedir. Gelişmiş bir web CBS'de Web Feature Service (WFS) olarak adlandırılan Web özellik servisleri, GML veri formatının elde edilmesi için gerekli olan bir teknolojidir. Bu teknolojilerin yanında, Genişletilebilir Stil Dönüştürme Dili (Extensible Stylesheet Language Transformation-XSLT) kullanılarak web tabanlı bir

coğrafi bilgi sistemi serbest olarak geliştirilebilir. GML verilerinin görüntülenmesi için Ölçeklenebilir Vektor Grafik biçimine (Scalable Vector Graphics-SVG) formatına dönüştürülmesini sağlamak amacıyla XSLT kodu geliştirilmiştir. Coğrafi veriler farklı XSLT stil dosyalarına göre farklı görünümlere sahip olabilmektedirler. Çalışma mekanizması olarak öncelikle GML ile hazırlanmış olan coğrafi veri dosyası, WFS yardımıyla gerçek zamanlı olarak elde edilir. Daha sonra, XSLT dönüştürme teknolojisi ile SVG formatına dönüştürme sağlanır. Bu noktada CSS stil kazandırma teknolojisinden de yararlanılarak katmanlara görüntüleme özellikleri kazandırılabilir. Bu şekilde, bir GML veri dosyası, tarayıcı yazılımı üzerinde yüksek kaliteli bir sayısal harita olarak görüntülenmektedir. Bu tez kapsamında, .Net platformu üzerinde, C# programlama dili ile ve MapXtreme geliştirme aracı kullanılarak interaktif bir kullanıcı arayüzü ile hizmet veren bir web-CBS geliştirilmiştir.

Anahtar sözcükler: Görüntüleme, XSLT, GML, SVG, CSS

ACKNOWLEDGEMENTS

This study titled “Visualization GML Data with XSLT for Web Based GIS” and presented as Master’s Thesis by Yaşar University Graduate School of Natural and Applied Science Department of Computer Engineering. I would like to special thank to my supervisor Assoc. Prof. Dr. Murat KOMESLİ for his support, advising and help on my thesis. He has guided me with suggestions and opinions, making necessary data easier to obtain and for supporting me with valuable contributions during my thesis study.

I would also like to especially thank the Yaşar University Rectory and the Information Technologies Department for technical support and for their help and support in my study.

I would also like to thank, the İzmir Büyükşehir Belediyesi (Izmir City Council) for its help with my thesis study.

In addition, many thanks to my deceased father, my mother and my family for encouraging me to manage this study during my thesis. Without their support this study could not have been realized.

Buket AKILLI
İzmir, 2014

TEXT OF OATH

I declare and honestly confirm that my study, titled “Visualization GML Data with XSLT for Web Based GIS” and presented as a Master’s Thesis, has been written without applying to any assistance inconsistent with scientific ethics and traditions, that all sources from which I have benefited are listed in the bibliography, and that I have benefited from these sources by means of making references.

.../.../ 2014

Buket AKILLI

Signature

TABLE OF CONTENTS

	Page
ABSTRACT	iv
ÖZET	vi
ACKNOWLEDGEMENTS	viii
TEXT OF OATH	ix
TABLE OF CONTENTS	x
INDEX OF FIGURES	xiii
INDEX OF TABLES	xvi
INDEX OF SYMBOLS AND ABBREVIATIONS	xvii
1 INTRODUCTION	1
1.1 Subject of the Thesis	1
1.2 Problem Definition	4
1.3 Aim of the Thesis	5
2 Geographical Information Systems (GIS)	6
3 Extensible Markup Language (XML)	11
3.1 XML Namespaces	15
3.2 XML Schemas	16

4	Geography Markup Language (GML)	20
4.1	GML Schemas	28
5	Extensible Styleseet Language For Transformation (XSLT)	31
5.1	XSL-FO XSL Formating Objects	38
6	Scalable Vector Graphics (SVG)	40
6.1	Structure of SVG Vector Graphics	41
6.2	GML Documents to SVG Graphics	45
7	CSS Styling Language	47
7.1	Basic CSS Selectors	47
7.2	CSS Usage in SVG Files	49
7.3	CSS Properties	56
8	Client Side Scripting With Javascript and EcmaScript	58
9	Web Feature Services (WFS)	62
9.1	OGC Web Services	62
9.2	Web Map Services (WMS)	64
9.2.1	Web Map Services (WMS) Operations	64
9.3	Web Feature Services (WFS)	65
9.3.1	Web Feature Services (WFS) Operations	66

10	MAPXTREME Development Toolkit	68
11	Configuration Settings	69
12	Related Works	73
13	Application	75
13.1	WFS Client	76
13.2	Mapinfo Tab Files Section	81
14	Conclusion	91
	REFERENCES	92

INDEX OF FIGURES

Figure 2.1 Client in the Server-side operations	8
Figure 2.2 Three-tier architecture in GIS	9
Figure 3.1 XML Functionalty not defined.	Error! Bookmark
Figure 3.2.1 Role of XML Schema in Systems defined.	Error! Bookmark not
Figure 4.1 Simple GML Geometry Elements	21
Figure 4.1.1 Geometry Elements of Point defined.	Error! Bookmark not
Figure 4.1.2 Geometry Elements of Polygon defined.	Error! Bookmark not
Figure5.1 North Arrow with SVG Format	36
Figure 5.1.1 XSLT Processor Functionalty not defined.8	Error! Bookmark
Figure 5.1.2 Translating of GML to SVG	38
Figure 6.1.1.1 Circle Drawing with SVG	41
Figure 6.1.2.1Polyline Drawing with SVG	41
Figure 6.1.3.1Polygon Drawing with SVG	42
Figure 6.1.4.1Path Drawing with SVG	42

Figure 7.2.1. CSS Usage in SVG Files	50
Figure 9.1.1 The difference between a raster and vector map format	63
Figure 9.2.1.1 A jpeg formatted map	65
Figure 12.1 The Application Mainpage's view	73
Figure 12.1.1 WFS Client Connection	74
Figure 12.1.2 Adding a New Connection	74
Figure 12.1.3 World Capitals	75
Figure 12.1.4 Oceans returned from WFS	76
Figure 12.1.5 Layer Show/hide Property	77
Figure 12.1.6 CSS Customization Tool	78
Figure 12.1.7 Customized Shape with the Tool	79
Figure 12.2.1 Turkey Tab File Visualization	80
Figure 12.2.2 Selection of İZMİR on the Map	80
Figure 12.2.3 Turkey Map with Layers	81
Figure 12.2.4 Turkey Map without Layers	81
Figure 12.2.5 Labeling on the Map	82
Figure 12.2.6 Theme options	82
Figure 12.2.7 Regional Theme related to Turkey	82

Figure 12.2.8 Pie Theme related to Turkey	83
Figure 12.2.9 Bar Theme related to Turkey	83
Figure 12.2.10 Graduated Theme related to Turkey	84
Figure 12.2.11 Update Panel	84
Figure 12.2.12 GML File Load Panel	85
Figure 12.2.13 Blank Turkey GML visualization	85
Figure 12.2.14 Turkey Provinces Map	86
Figure 12.2.14 Turkey Provinces Map with Label	87
Figure 12.2.15 Provinces of Turkey	87
Figure 12.2.16 Ardahan GML file visualization	88

INDEX OF TABLES

Table 4. 1. SVG Correspondences of GML Geomety Elemenets	24
Table 5.1 XPath Expressions	35

INDEX OF SYMBOLS AND ABBREVIATIONS

ASCII	American Standard Code for Information Interchange
API	Application Program Interface
CSS	Cascading Style Sheet
CBS	Coğrafi Bilgi Sistemleri
DTD	Document Type Definition
DSSSL	Document Style Semantics and Specification Language
GML	Geographic Markup Language
GIS	Geography Information System
HTML	Hyper Text Markup Language
HTTP	Hypertext Transfer Protocol
OGC	Open Geospatial Consortium
RDF	Resources Description Framework
SVG	Scalable Vector Graphics
URL	Uniform Resource Locator
WFS	Web Feature Service
WMS	Web Map Service
WCS	Web Coverage Service

1 INTRODUCTION

1.1 Subject of the Thesis

Geographical Information Systems, abbreviated as GIS, have a variety of different definitions and interpretations according to variety of different disciplines. One of the definitions of Geographical Information Systems (GIS) among these definitions is “It is an information system that makes spatial-based information which is obtained with observations and non-spatial information, collecting, storing, presentations to the user as completeness”. The history of GIS originates in 1960. In those years GIS was used for special tasks for the military in the USA and Canada. But after 1980, it started to be used for commercial areas, becoming more widespread. Besides this, in recent years, the internet has become popular rapidly and as a result Web-based Geography information systems have also become developed for this popularity. Web-based Geography Information Systems are growing day by day. In Geography Information Systems any information which enables to give reference information to any location is named as Spatial Information. Spatial information is a kind of information which obtains data for users related with map structure. For the reason being that the web/internet is more popular, Information Systems become a concept which is used among various disciplines. It has started to handle using and administration of spatial information which is needed for various disciplines together with information systems. In this process, it is known that 80% of information is based on location information. For the reason of importance of the spatial information, it becomes more important to represent world surface the most near to real. In addition, it is needed to standardize geographical information which is used in applications. These advances have started to be assimilated by corporations and have contributed to the establishment of interoperability. Web-based Geographical Information Systems have started to become important areas of projects which are performed by software development corporations. The internet is preferred for the reason of ease of the presentation of the Geographical Information Systems and usage of this kind of information. In addition, thanks to the internet, it is not an easy development of open source coded projects.

Data Source of a Geographical Information Systems can be handled as “category criteria” which is used in the GIS industry. It can be shown that Geographic Markup Language (GML) is a source type for these types of applications. For this

reason, GML source data type is the type of data of this piece of research. With the second version of the GML geography markup language, a foundation was laid for the spatial-locational internet on 20 February 2001 by the Open Geographical Consortium (OGC) (Dempsey, 2001). The Open Geographical Consortium (OGC) was established in 1994. It is an industry community which works with ISO/TC 211. The aim of this community is to specify standards related to locational informations applicable (Aras and Yıldız, 2011). The Vision of the community is that it is provided to construct a platform, an application or a network which is of benefit for all users who need or use spatial and locational information (Aras and Yıldız, 2011). The mission of the community is that spatial/locational interface and coding technic standards is rendered to open situation for all users (Aras and Yıldız, 2011). GML markup language is Open Geography Information Standards community which is published from OGC (Open Geospatial Consortium) and one of the base components of national geography data infrastructure. Being the same as all the other ISO standards, translation of Geography Information Standard to national standards and distribution of these standards in Turkey is up to “Türk Standartları Enstitüsü” (TSE) (Wikipedia, 2010). GML is published by Open Geospatial Consortium (OGC) and is based on XML Extensible Markup Language (XML). GML has been developed for interoperability problems for a long time now. Presently, the Geography Markup Language is used for obtaining transportability of spatial /locational information among platforms and storing this information. In GML (Geography Markup Language) documents, Geographical data is stored as isolated from presentation elements because of XML based technology. With these properties, it is possible to get a GML (Geography Markup Language) document’s different visualizations on web browsers. GML (Geography Markup Language) data file’s visualization technique is the same as XML (Extensible Markup Language).

It is known that interpretation of the document which is prepared with XML (Extensible Markup Language) is made with XSLT (Extensible StyleSheet Translation Technology) and visualization of this XML based document, XML document, is made with SVG (Scalable Vector Graphics) on the web browser. GML data is visualized in SVG to provide high-performance and is possible with the XSLT translating mechanism. (Keeneth S.Herdy and others, 2008) At the same time, produced SVG documents are styled with CSS. Interactivity properties are made with JavaScript and EcmaScript scripting languages.

Graphical representation is necessary for visualization of any location. The most preferable method/technology for visualization of XML based data is SVG (Scalable Vector Graphics) technology. This technology produces scalable graphical vectors. SVG (Scalable Vector Graphics) technology is used for lack of producing this type of visualization in HTML. XML technology works with embedded HTML tags mechanism which is located in HTML technology for web browsers. SVG (Scalable Vector Graphics) technology is a technology which constructs dynamic graphics. It manages to visualize spatial/geographical objects the most effectively. It is possible for these mentioned web technologies to be implemented for web-based systems. SVG vector graphics is the most preferable technology for the following reasons. The first reason is that it is necessary for possible minimum bandwidth size. Another reason is the fast downloading to other web technologies. These reasons make SVG the most preferable. GML (Geography Markup Language) and SVG (Scalable Vector Graphics) have the same structure and format. For this reason it is possible to get SVG vector graphics from GML data (Geography Markup Language) with XSLT (Extensible StyleSheet Translation Technology) transformation files. It is possible when XSLT transformation files are applied to one more type of GML files. Then, XSLT becomes an automation for this visualization of GML files.

CSS Commands can be included externally which is defined in XSLT (Extensible Stylesheet Translation Technology) translation file by defining in an external CSS file. It should be noted make that internally defined CSS Commands are more prior than externally defined CSS commands. In externally defined CSS commands, there are some properties like that it splits style files from style elements. Thus, it can be possible for externally defined CSS commands that it is valid for all GML (Geography Markup Language) files. In addition it is possible visualize the same GML file with different stylesheet files. With SVG vector graphics, which are obtained from XSLT translation files, it is possible to apply various platforms like a web page or an XML file.

There are 10 sections in the scope of this thesis. The first sections, “Geography Information Systems (GIS)” and “Web based Geographical Information Systems” are handled as a topic. In section two, “XML Extensible Markup Language” is mentioned. In section three, the “Geography Markup Language (GML)” is dealt with. Section four, covers the “XSLT (Extensible Markup Language) Translation Technology”. In section five, “SVG (Scalable Vector Graphics)” is described. In

section six, “Styling with CSS (Cascading Style Sheet)” is explained and then some information about the topic is given. In section seven, a scripting language, JavaScript, is explained. In Section eight, “WFS Services” for obtaining GML file from various servers is covered. In section nine, “MapXtreme Development Kit” it is mentioned and explained. In section ten, configuration settings are mentioned as a topic. In section eleven, application is introduced. In section twelve, the advantages of these technologies used is discussed.

1.2 Problem Definition

The starting point of the thesis is Visualizing the most preferred geographic data format document, GML, at Geography Information Systems (GIS) on the web platform. In addition, it is needed to pass in front of the commercial software which is in front of the obstacle for visualizing spatial information. At the same time, it is purposed to overcome of the interoperability problem with using GML spatial data files including web technologies. Geography Markup Language (GML) which provides the standardization of the system data files are not a presentation document in other words, it is not a structure via the web browser, users can view. For this reason, this GML file is necessary to translate automatically to presentation file with the help of a technology for visualization. This comprises one of the sections of the problem.

Individual addressing of the geometry of the elements and topologies which are located in GML documents for this automatic translation. Besides this, it is intended to access smart vector format maps which is high-resolution, scalable with SVG (Scalable Vector Graphics) web technology, in order to improve the quality of the produced maps. Another section of the problem is that, the mentioned GML files sizes are very large. The problem is that the performance which is necessary for interpretation on the web browser. The main problem in this thesis is confirming what prevents some of the disadvantages of the internet. Thus, together with the thesis, it will have a “Web Map Service” which performs freely representations of maps that are located on the user's own computer or connected to a preferred service provider.

Nowadays, Web services (WS) is a technology that has active application areas according industrial and technology. It has shown the same trend in demand in the GIS industry. As a result, it is becoming possible to access the maps with WFS (Web

Feature Service) and WMS (Web Map Service) services. The scope of this thesis is that the preferred WFS services which can cover GML (Geography Markup Language) data presentation from web services.

1.3 Aim of the Thesis

The aim of this thesis is to determine the strategy for visualizing GML geographical data standard which is one of the common development areas in and academic and industrial areas and is a standard of Geographical Information Systems (GIS) on the web browser with web technologies and standards.

In the realization of this study, it was deduced with the examination of the web technologies that can be made compatible with each other and problems are analyzed in traditional models and Web services are can be reviewed and applications were made for each of the inspected web technologies in practice and then it was overcame determination of the technical implementation strategy by combining all distributed technologies.

2 Geographical Information Systems (GIS)

Geographical Information Systems (GIS) are an information system that obtains storage, manipulating, visualizing, querying, analyzing of geographical information and other in information related with geographical information. Geographical Information Systems (GIS) are interdisciplinary systems consisting of a merger of the two disciplines which are Geography and Information Systems. Geographical Information Systems (GIS) are special systems that bring together the power of a relational data base for use and the Visual interface of maps. The relationship between spatial/geographic information and other information related with geographic information which belongs to any location on earth can easily be incorporated with the help of this system. In the early days of the Geographic Information Systems (GIS), there was a computer and the data sets located on the same computer. In a globalized world, this situation will be a problem for enterprises. On the other hand, the biggest obstacle in front of the Geographic Information Systems was the capability of a system to interact problem and interoperability. In the 1990's, There had been significant impacts on the geographic information systems for the rapid spread of the internet. These are "Data Access", "Data Transmission", "Accessing Geographic information systems (GIS) analysis functions ". The benefit of the Geographic Information Systems (GIS) regarding accessing the analysis functions is that the user can access and process the remote computer with just using internet a browser only. The internet was first introduced in 1969 for protecting the information related with military in a time of war. Since 1983, the internet has started to be used as a TCP/IP supported link. For this reason, the internet has become an appropriate platform for sharing geographical information with represented facilities. Information is obtained from data in Geographical Information Systems. An information system resumes thanks to data which constitutes the information. In Geography Information Systems (GIS), the data gathering phase is the most time consuming and requires the maximum cost phase. Because of this, Geography Information System (GIS) workers adopt an approach that obtaining the data obtained from other sources before, instead of obtaining the data from scratch and converting your own GIS format for lessening costs. At this point, displaying in web browsers to convert GML formatted data to display format serves to "data conversion and visualizing" which is the most important part of the Geographic Information Systems (GIS).

More effective Geographic Information System applications have been developed by utilizing Web technologies, but there is no Geographic Information System application which serves more than one platform, the discipline and the user. This situation has forced developers to revise the adopted GIS system approaches and to create a new State-of-the-art integrated system. Web based Geographical Information Systems have been developed under of this scope and use of web technologies. These web technologies are GML (Geographic Markup Language), XSLT (Extensible Style sheet Language Transformation) transformation language, SVG (Scalable Vector Graphics) vector graphic language and WFS (Web Feature Services) services. To understand a web-based Geographical Information System (GIS) working mechanism, it is necessary to know the principles at work.

In the architecture of the client and server, the client and server connect by using the HTTP protocol on the intranet or internet in TCP/IP (Transmission Control Protocol/ Internet Protocol) based networks. (Aydinoğlu A.Ç., 2003). With the help of a web browser, a command is sent to the server by the client on CBS. (Aydinoğlu A.Ç., 2003). The answer which is produced by the client in the server-side operations result is sent back through the URL (Uniform Resource Locator) addressing. (Aydinoğlu A.Ç., 2003). In Figure 2.1 this scenario is described.

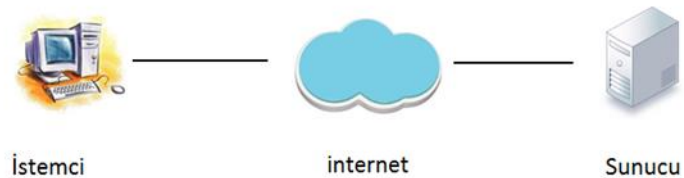


Figure 2.1 Client in the Server-side operations

Software functions which are used in the Geographic information systems like “Accessing spatial data and analysis functions”, ”geographical analysis”, “viewing analysis results” operations are located both traditional Geography Information Systems and web-based Geography Information Systems. There are a number of properties related with web based Geography Information Systems.

Because of web based geographical information systems being structured as client/server architecture, when analyzing, operations are distributed between the

server and the client. Firstly, the client requests the data and analysis function from the server. And then the server sends the request to the client. Web-based Geographical Information System (web GIS) are distributed and dynamic systems. Because of this, all data and applications are located on the server and the client only makes of using the system.

In classic models of web-based geographic information systems, there are two components. These components are client and server. But for three-tier web-based GIS systems, there are three components. These are client, GIS (Geographical Information System) web server and server. This thesis also adopts the three-tier architecture. In Figure 2.2 this scenario is described.

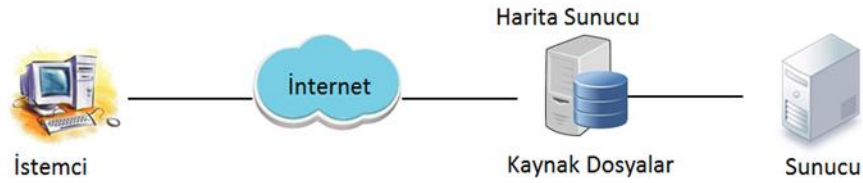


Figure 2.2 Three-tier architecture in GIS.

There are differences between web based geography information systems and standard GIS systems. These are storage and processing of the information and user interface. There are a lot of advantages of web-based geographical information systems to other standard geographical information systems. These advantages are explained below;

-Ease of Access: Users can reach easily the system just using the internet from anywhere in the world via a Web browser with no restriction and without the need for any other 3rd party software. This feature led to elimination of the discrepancy between data.

-Ease of Use: Users who do not have knowledge about this system can easily use the system.

-Standard Interface: All of the users can easily access a single standard system interface because of depending on a single service provider.

-Dynamic Maps: Works with dynamic maps which query, analyze and communicate the databases with the introduction of the internet.

-Storage Costs: Thanks to the internet, the user doesn't need store the maps to work on. They can access the system through the internet when they want.

-Economic and fast Maintenance: Because users access information from only one source when have any problems or when updating service, provider (server) operations are arranged only with one service provider and then the service to the users is provided simultaneously. In other words, update, maintenance and repair operations are made separately for each user.

Here it can be seen that Geographical Information systems (GIS) have an important role for using geographical information. On the other hand, the internet is important for sharing this information which is used from the GIS system. In web-based Geographical Information systems users can easily reach maps which they want over the internet. At the same time, users can save these maps to their own computers. Thus, they can have the opportunity to work on them. When developed, a web system which works on the internet has a lot of components which determine the type and character of the system. These are the methodologies which are dependent on the users' requests, software architectures, the technologies and methods to be used. The characteristic features of the data for determining which technologies will be used for the web based geographical information systems' applications which are to be developed are taken as the benchmark. It is possible to come across 3 different strategies. These are "Server-Side" strategy, "Client-Side" strategy, "Hybrid" strategy. In "Server Side" strategies, the client sends the request to the server and the map is sent back to the client as a response after processing the request on the server side. In "Client Side" strategies, evaluating the request sent by the client is made on the client's computer. In "Hybrid" strategy, the relationship between them will continue after send the request to the server.

Besides these, the mentioned advantages of web-based geographic information systems, data retention and backup operations are issues that should be considered. Due to the large size data, additional backup areas are needed. It is also important that

sharing of collected information, besides that of data collection during GIS (Geography Information System) development. There are some problems during transfer of backed up geographical information to the web browser for visualization. These problems are insufficient bandwidth and downtime issues. These problems are the difficulties of working with large scale data.

3 Extensible Markup Language (XML)

Because communication has become an indispensable part of our lives, a common language which is available to all of the people is needed. For this reason, a common language was sought. English, which is the most commonly used in the world of technology and the internet has become the common language in the greater world. Based on the same sample, XML (Extensible Markup Language) technology is a language which is the most appropriate standard language for communicating between the different systems (an incompatible) in the Information Technology World. The purpose of naming as Extensible is that it is a structure which defines analyses and reshaping the data as a technology as a result it is a technology which prevents shortcomings of HTML. Today, XML technology which is software and hardware-independent markup language is used as a packing service while transferring data between applications and platforms. In addition it works on the tasks that running in the applications background. In Geographical Information Technologies, XML is most important technology for querying processing. It is known that XML transferring occurs during the client/server relationship among all GIS systems. As can be seen below, in Figure 3.1, there is a showcase of this processing.



Figure 3.1 XML Functionalty

XML (Extensible Markup Language) is a data format which is published freely from W3C consortium. W3C consortium is a community that publishes all web standards. XML (Extensible Markup Language) technology is derived from GML (Standard Generalized Markup Language) technology. XML which is very flexible structure is a markup language, in fact, has emerged for use in the field of Electronics. XML Extensible Markup Language has a wide range of areas. Now,

XML (Extensible Markup Language) is important for transportation of large scale data on the web platform. Today, XML is a format which is supported by many applications. The most important role of the XML technology is that it is a necessary technology for interoperability problem. So much so that, once it is created, an XML (Extensible Markup Language) file is a source for more than one application. There is XML support for almost all programming languages. At the same time, all database systems support the XML technology. In XML technology, data is defined with metadata. Metadata is the definition of data, in other words, it is said that data definition about the data.

In Geography Information Systems (GIS), XML technology is important for querying processes. It is possible to work and access the web services with using XML technology. Geographical information which is written in XML markup language is send to outside world by using a stream.

The character of XML (Extensible Markup Language) is the same as HTML (Hyper Text Markup Language). Both of them are text-based languages. XML data format defines data as sequential alignment with one or more elements.

If we examine the structure of the XML, there are two components. First is the title which includes some of the required information for the parser and elements. The second component is that of XML schemas which validate XML documents and include a set of rules inside XML documents. XML schema's previously used DTD (Document Type Definition). XML documents are located in web browser and embedded. In terms of the meaning of the XML technology, it can be expressed as encoded information in the text. Differences between GML and HTML are that it is not necessary previously defined labels and users do tag definitions in a manner appropriate to the rules. Due to these features, it can be said that XML markup language is highly flexible and extendible. In addition, XML is easy to understand because of ASCII (American Standard Code for Information) based by people and computers. (LIANG ZOU, 2004).

Because of text-based technology, XML can be combined with a wide range of data types which contain text. Because of this, it is possible to store easily in the same file geographical information and other non-geographic information which dependent

to geographic data. This situation is a technique that access geographical/spatial information.

There are two methods for XML (Extensible Markup Language) operation. First of these is DOM (Document Object Model) streaming model, second of these is SAX3 which is a basic API (Application Program Interface) for using XML (Extensible Markup Language).

XML (Extensible Markup Language) processors handle XML documents in the memory with taking as DOM (Document Object Model) tree. When it is considered a large XML documents, DOM is insufficient because it requires internal memory space and time. On the contrary, in the streaming model, the XML data document is displayed as a data stream. One of the great advantages of using the streaming model is that it requires more economic footprint in the memory. However, in the streaming model, unlike the DOM (Document Object Model), it is not possible that users' querying operations are worked directly and random access to documents. Performance issues are important because the aforementioned is Geographical Information System (GIS) which uses geographical data.

Below there is an example which is a short road segment GML encoding that showcases a road segment and also showcases XML structure's characteristic properties. Attention should be made here that the data is kept together and at the same time there is a hierarchical approach in the relationships to each other.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<uka:Road fid="highway30">
  <uka:numLanes>5</uka:numLanes>
  <uka:surfaceType>gravel</uka:surfaceType>
  <gml:centerLineOf>
    <gml:LineString srsName="epsg4361">
      <gml:coordinates> .... </gml:coordinates>
    </gml:LineString>
  </gml:centerLineOf>
</uka:Road>
```

If we talk about XML's writing rules, there is an element of an XML document that is just a "root" element. Tag and attribute names are "case sensitive". Each opened tag has a closing tag. There is an important point is that there is no intersection in the same named tags. It should be written attributes with taken quotes

in XML documents. XML document which include all of these requirements named as “well formed” XML document.

There are various kinds of XML editors like notepad or other improved XML editors. These improved XML editors provides the convenience of preparation. Users define two different types for preparing XML documents. The first is that the user makes a definition of tag pairs which is including the opening and closing tags. Second is that user enters tags content. We can come across many different places with XML technology. Because, XML technology has extensive uses of area. For example, it is possible that XML technology consists with new programming languages thanks to its flexible and understandable structure and easy creation and processing and ease of wide area of usage of XML technologies. GML is also an example of these constructed languages. In this section, XML documents were analyzed rapidly and shortly, now, general structures of GML documents are handled.

In the scope of the thesis, there is a developed application. The application is based on .Net platform. It is known that .Net platform uses XML technology in widespread area and it presents rich features for make operations on the XML data. For example, there is an XML address file which holds web service name inside and previously entered on the user’s account. In this file, there are an URL path of WFS service and a name which is defined by user for this service. When it is wanted to add a new address information to this file, it is required to open the XML file and then it is needed to save information which is wanted to be saved. Below, there is an example of code which exists in the project. These codes manage to read and write addresses of web services to the screen.

```
if (!File.Exists(path))
{
    FileStream fs = File.Create(path);
    fs.Close();

    FileStream Servicexml = new FileStream(path, FileMode.Create);
    XmlTextWriter xml = new XmlTextWriter(Servicexml, Encoding.UTF8);

    xml.WriteStartDocument();

    xml.WriteStartElement("WfsAddress");
    xml.WriteEndElement();
    xml.Close();
```

```
        return;  
    }
```

Another example is that there is a code sample which is written with C# language on the .Net platform. The code manages XML serialization. So, registration a new WFS address is managed to the XML document.

```
XmlDocument doc = new XmlDocument();  
if (File.Exists(path))  
{  
    doc.Load(path);  
    XmlElement address = doc.CreateElement("Address");  
  
    XmlElement wfsName = doc.CreateElement("WfsName");  
    XmlElement urlName = doc.CreateElement("URLName");  
    wfsName.InnerText = WfsName;  
    urlName.InnerText = URLName;  
  
    address.AppendChild(wfsName);  
    address.AppendChild(urlName);  
  
    doc.DocumentElement.AppendChild(address);  
    doc.Save(path);  
}
```

3.1 XML Namespaces

XML Namespaces prevent the meaning complexity between the elements in the XML documents. Often, in XML technology, conflict problems may occur from the user-side encoding when included in another XML document because tags and content information are constructed by the user. Each element's prefix as defined in the namespace prefix distinguishes created elements which are constructed for different purposes of the same names. Each geographic markup language (GML) document must contain "Namespaces" as follows.

```
<ogr:FeatureCollection  
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
xsi:schemaLocation="C:\Users\Buket\Desktop\Turkey Turkey.xsd"  
xmlns:ogr="http://ogr.maptools.org/"  
xmlns:gml="http://www.opengis.net/gml">
```


3.2 XML Schemas

XML schema documents are documents which are published by the W3C consortium. XML schema documents give information about the XML document structure, attributes and the semantics of the document. XML schema documents explain how an XML file is marked. XML schema documents are saved with the .xsd extension. XML schema documents make validation of XML files when the documents are included in the XML files. XML schema documents are written in XML language. So, it is in full compliance with other xml technologies.

Before the XML schemas, the DTD (Document Type Definition) file was used for schema definition language. DTD (Document Type Definition) file defined which property would be elements and attributes in a GML document. XML documents which are written with rules specified in the DTD (Document Type Definition) were valid documents. But, there were many restrictions in DTD technology. One of the restrictions was that it does not support data types which are used on the databases. Another restriction was very limited data type support. One last restriction was that it was not suitable for XML spelling rules. Now, XML schemas are used for validating operations. XML schemas are the technology that has been developed to resolve the shortcomings of the DTD. The biggest advantage is that it supports data types.

XML schema files are included externally to XML files as a validating file. An XML schema file defines data types of the elements in an XML document. There are a lot of rules inside the XML schema document. When find an element which does not suit any of these rules in the XML document, it is considered an invalid XML document. There is a root element in XML schemas and expressed with <schema> tag. There are two kinds of element type. These are “Simple Element” and “Complex Element”. Simple Elements just contain a value. Simple Elements don’t contain an attribute or any other element. In addition, these elements are never empty. If an element isn’t simple, it is a complex element. So, a complex element contains attributes or any other elements. In addition, these elements can be empty. Or, it can contain both at the same time. An xml schema which belongs a GML file is defined as follows.

```
<ogr:FeatureCollection
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="C:\Users\Buket\Desktop\Turkey Turkey.xsd"
```

In the section which is defined as “schemaLocation”, first file path and second file name is described. It must be a blank between “file path” and “file name”. So that, with this operation, XML schema is included to GML document externally. Below, in Figure 3.2.1, XML schema’s role among the other XML technologies is defined.

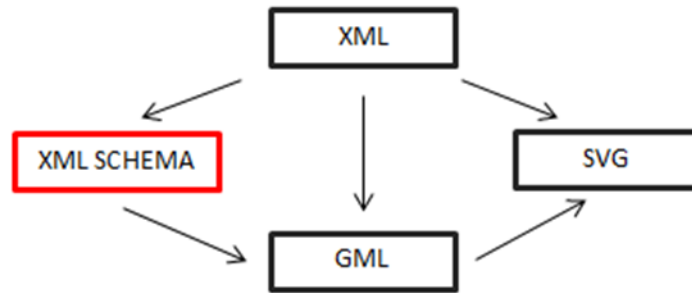


Figure 3.2.1 Role of XML Schema in Systems

In Figure 3.2.1, above, it can be seen that XML (Extensible Markup Language) schema and GML (Geography Markup Language) technologies together have a logic that allows the transformation to SVG document (Scalable Vector Graphics). There are XML schemas within GML, “feature.xsd” which is served by Open GIS Consortium, “geometry.xsd” and “xlink.xsd” which is served by W3C consortium. This creates "application Schema" when schemas come together. The XSLT file is created according to the application Schema. Within a schema document, it is made namespace and schema definitions in the root index like at the below.

```

<xs:schema targetNamespace="http://ogr.maptools.org/"
xmlns:ogr="http://ogr.maptools.org/"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:gml="http://www.opengis.net/gml" elementFormDefault="qualified"
version="1.0">
<xs:import namespace="http://www.opengis.net/gml"
schemaLocation="http://schemas.opengis.net/gml/2.1.2/feature.xsd"/>

```

As we see above, each root element and other elements are created with taking xsd prefix. After that, it is made schema definition which belongs to the reference XML documents. Below, more sample code can be seen:

```

<xs:element name="Turkey" type="ogr:Turkey_Type" substitutionGroup="gml:_Feature"/>
<xs:complexType name="Turkey_Type">
  <xs:complexContent>
    <xs:extension base="gml:AbstractFeatureType">
      <xs:sequence>
        <xs:element name="geometryProperty" type="gml:PolygonPropertyType" nillable="true"
minOccurs="0" maxOccurs="1"/>
        <xs:element name="IL_ADI" nillable="true" minOccurs="0" maxOccurs="1">
          <xs:simpleType>
            <xs:restriction base="xs:string">
              <xs:maxLength value="20"/>
            </xs:restriction>
          </xs:simpleType>
        </xs:element>
        <xs:element name="PLAKA_NO" nillable="true" minOccurs="0" maxOccurs="1">
          <xs:simpleType>
            <xs:restriction base="xs:string">
              <xs:maxLength value="2"/>
            </xs:restriction>
          </xs:simpleType>
        </xs:element>
        <xs:element name="NUFUS_1990" nillable="true" minOccurs="0" maxOccurs="1">
          <xs:simpleType>
            <xs:restriction base="xs:integer">
              <xs:totalDigits value="10"/>
            </xs:restriction>
          </xs:simpleType>
        </xs:element>
        <xs:element name="NUFUS_1997" nillable="true" minOccurs="0" maxOccurs="1">
          <xs:simpleType>
            <xs:restriction base="xs:integer">
              <xs:totalDigits value="10"/>
            </xs:restriction>
          </xs:simpleType>
        </xs:element>
        <xs:element name="DEGISIM" nillable="true" minOccurs="0" maxOccurs="1">
          <xs:simpleType>
            <xs:restriction base="xs:decimal">
              </xs:restriction>
            </xs:simpleType>
          </xs:element>
        <xs:element name="ORT_GELIR_1980" nillable="true" minOccurs="0" maxOccurs="1">
          <xs:simpleType>
            <xs:restriction base="xs:decimal">
              </xs:restriction>
            </xs:simpleType>
          </xs:element>
        <xs:element name="Yuzolcumu" nillable="true" minOccurs="0" maxOccurs="1">
          <xs:simpleType>

```

```
<xs:restriction base="xs:decimal">  
  <xs:totalDigits value="6"/>  
  <xs:fractionDigits value="3"/>  
</xs:restriction>  
</xs:simpleType>  
</xs:element>  
</xs:sequence>  
</xs:extension>  
</xs:complexContent></xs:complexType>
```

4 Geography Markup Language (GML)

Geography Markup Language (GML) is based on Extensible Markup Language (XML) and is published from Open Geospatial Consortium (OGC). GML markup language is developed for definition, storing, transporting of spatial/geographical information. In addition, it is developed for interoperability problems among systems. GML is also considered as transporting of data. GML is a standard which is derived from XML. In addition, it is a data format. Geographical information is stored in GML format and is translated to wanted format when it is necessary with using web technologies. GML which provides exchange of data between the various platforms has been prepared according to the rules. (OGC Standards, 2014)

In the first years of GML, it was modeled by RDF (Resource Description Framework) modelling technique. RDF is a technology that defines the source of the World Wide Web (WWW). After that, OGC defines XML schemas inside the GML documents. So, it enables easy communication with a variety of geographic databases. (JIANG Jun et al., 2008).

GML is instantiated for web technologies and web based services. GML is prepared according to “Geographical Information Systems Specifications”. GML makes spatial data exchange among various platforms. A GML document contains spatial and non-spatial information which is related with spatial information. A GML document is used for not only GIS systems but also geographical database as a data file. In the address of “<http://www.opengis.org/technospecs.htm>”, it is explained that how translate GML files real world to two dimensional demonstration. (Komesli and Ünalır, 2004).

All the spatial details are obtained from consideration of the earth’s surface in abstract form. In positional detail, each of the details is defined as a set of names, types and value properties. An example can be seen below.

detailX : { isim, tip, değer }

Beside of name a type of a detail, also the number of properties that it can have are indicated with identification of the type. (Komesli and Ünalır, 2004). GML

digitizes the real world and then gets two dimensional models of the real world. So, GML creates a collection of the details. For example, it is defined “Name” property in the city detail. Besides basic properties like the “Name” property, there are also properties which give information related with geometry. For example, in the city detail, there are geometric properties such as “Polygon”.

As a structure of GML, GML includes in itself both a geometry element and properties of geometry elements for coding spatial information in the spatial data file. GML has many objects within documents to identify *geographical features*. These objects are like this. “Coordinate Reference Systems”, “Geometry”, “Topology” ”Time”, “Units of Measure”, “Generalized Values”. In addition, there are basic geographical objects for demonstration of real world as two-dimensional. These are “Point”, “MultiPoint”, “Polygon”, ”Multipolygon”, ”LineerRing”, “MultiGeometry”.

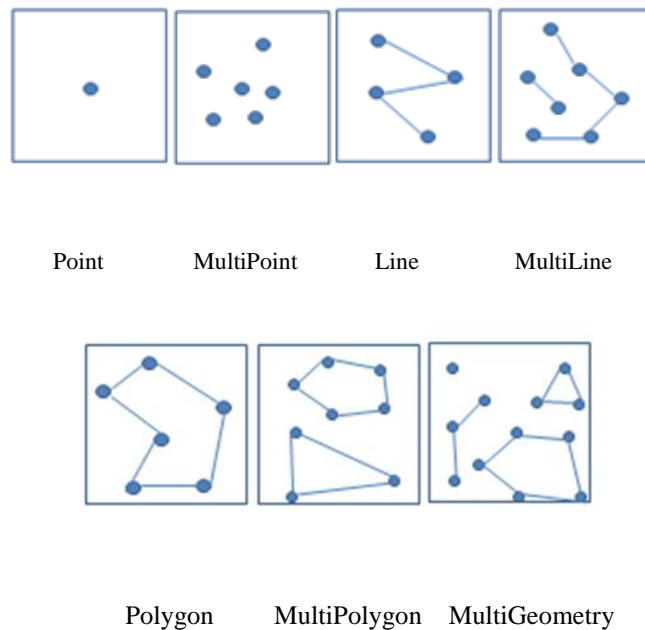


Figure 4.1 Simple GML Geometry Elements

In addition to the elements of this geometry, there are a box elements which define identified geographical area’s boundaries and <coordinates>, <coord>, <pos> elements which define coordinates information related with geographical information. This diversity can change according to version of GML and geometry elements. It is

used as different varieties in different geometries. For example, `<coord> ... </coord>` is a geometric detail which is used in Point object. Usage of these tags are like this.

```
<coord><X> </X><Y> </Y></coord>
```

If we give an example of `<coordinates>` tag, it can be given “Polygon” tag.

```
<gml:Polygon><gml:outerBoundaryIs><gml:LinearRing>
```

```
<gml:coordinates> Noktalar kümesi </gml:coordinates>
```

```
</gml: LinearRing>gml: outerBoundaryIs><gml: Polygon>
```

There are static properties in each GML document which contain geographical data. Besides this, GML data’s modelling logic and labelling structure according to users/developers coding technique changes. Every GML document starts with “XML standard Caption”. After that, the “CSS (Cascading Style Sheet)” and address are defined. In the root of GML documents, there are schema location definition and namespace definition. Below, there is a showcase of basic GML documents.

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="http://www.buketakilli.com/gml/landsacpe.xsl"?>
<gml:FeatureCollection
  xmlns:gml="http://www.opengis.net/gml"
  xmlns:xlink="http://www.w3.org/1999/xlink"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:fme="http://www.buketakilli.com/gml"
  xsi:schemaLocation="http://www.buketakilli.com/gml/landsacpe.xsd">      ( Section 1 )
```

Below, there is a small GML (Geography Markup Language) example related with “boundedBy” property. In this example, the boundary value identification of geographical information which is located in GML document is defined. This definition mentioned above is made after all namespace, schema, and style definitions.

```
<gml:boundedBy>
```

```

    <gml:Box srsName="GHTR:984">
      <gml:coordinates>170000, 436000, 439000</gml:coordinates>
    </gml:Box>
  </gml:boundedBy>

```

(Section 2)

After this definition, it is followed with properties hierarchy which is defined by “GIS Specification Rules”.

```

    <gml:featureMember>
      <ogr:Turkey fid="Turkey.1">
        <ogr:geometryProperty> <gml:Polygon> ... </gml:Polygon>
      </ogr:geometryProperty>
      </ogr:Turkey fid="Turkey.1">

      <ogr:Turkey fid="Turkey.2"> . . .

      .
      .
      .

    </ogr:Turkey fid="Turkey.2"> ... </gml:featureMember>

```

(Section 3)

GML contains geographical information inside. Because of that geometry elements are defined according to geographical properties of markup language. Some of these properties are named as Basic Properties of OGC (Open Geospatial Consortium). As we mentioned, these properties are Point, LineString, LinearRing, Polygon, MultiPoint, MultiLineString, MultiPolygon, MultiGeometry and Box. If we want to group, there are three groups. These groups are “Point”, “Line” and “Polygon”. There are some special definition rules for defining GML data. One of these rules is that it is taken a “gml” prefix which is identified an element belonging to the GML data during defining the geometry elements. This prefix comes from `xmlns:gml=http://www.opengis.net/gml` namespace. It is used every time “gml” prefix structure for this namespace. After GML namespace is included in a document, all geometries which is prepared from OpenGIS can be used within a GML document. For example a Point is defined with gml prefix like the one below.


```

<gml:Point>
  <gml:coord>
    <gml:X>5.0</gml:X>
    <gml:Y>20.0</gml:Y>
  </gml:coord>
</gml:Point>

```

Another GML definition property is that it mustn't be taken as part of a Feature Collections element within the Feature the collections element which is defined in GML document. gml: featureMember element in GML document represents a feature which is inside a Feature Collection. GML files are always written with UTF8 encoding. The geometries are shown in the following table. In the table, each of the geometries which correspond for SVG technology in the future is included.

GML	SVG
<Point>	It can be represented by a rectangle, a circle or a icon.<svg:point>, <svg:rect>
<LineString>	<svg:polyline>
<LinearRing>	<svg:polygon>
<Polygon>	<svg:polygon>
<Box>	<svg:rect>
<MultiPoint>	It can be represented by a rectangle, a circle or a icon.<svg:point>, <svg:rect>
<MultiLineString>	It is defined inside <g> element as a group of <svg:polyline> element.
<MultiPolygon>	It is defined inside <g> element as a group of <svg:polygon> element.

Table 4. 1. SVG Correspondences of GML Geometry Elements

It is defined nodes which are defined between GML geometry element as "Property". There are also elements which act as geometry elements. These elements are named as "Convience". There are a lot of elements which is "Convience". These are "BoundedBy", "CenterOf & "Position", "EdgeOf", "CenterLineOf" and "extendOf".

The following example is based on some simple geometry definitions according to version and geometry.

Point Example

```
<gml:Point>
  <gml:pos>0 100</gml:pos>
</gml:Point>
<gml:Point>
  <gml:coordinates> 0,100
</gml:coordinates>
</gml:Point>
```

Polygon Example

```
<gml:Polygon>
  <gml:exterior>
    <gml:LinearRing>
      <gml:coordinates>.....</gml:coordinates>
    </gml:LinearRing>
  </gml:exterior>
</gml:Polygon>
```

Validation is provided by inclusion of “GML schema Documents” which provide the current document to be valid documents to GML documents. Schema documents are external documents for encoding with GML. These documents are “GML Detail Schema” (feature. xsd) and “GML Geometry Schema ” (geometry. xsd). With these schema documents, it is not necessary geometry for encoding geographical information. For example, a GML instance and GML schema definitions that belong to the GML instance can be given as follows.

```
<gml:featureMember>
  <ogr:Turkey>
    <ogr:IL_ADI>ADANA</ogr:IL_ADI>
    <ogr:PLAKA_NO>01</ogr:PLAKA_NO>
    <ogr:NUFUS_1990>1549233</ogr:NUFUS_1990>
    <ogr:NUFUS_1997>1682483</ogr:NUFUS_1997>
    <ogr:DEGISIM>11.6055524796825</ogr:DEGISIM>
    <ogr:ORT_GELIR_1980>497904000</ogr:ORT_GELIR_1980>
    <ogr:Yuzolcumu>14.256</ogr:Yuzolcumu>
  </ogr:Turkey>
</gml:featureMember>
(GML document)
```

```
<element name="Turkey " type="ex: TurkeyType" />
  <complexType name=" TurkeyType">
    <sequence>
```

```

    <element name= IL_ADI " type="string"/>
    <element name="PLAKA_NO " type="integer"/>
    <element name="NUFUS_1990" type=" integer " />
    <element name= NUFUS_1997" type=" integer" />
    <element name=" DEGISIM" type="integer"/>
    <element name=" ORT_GELIR_1980" type="integer"/>
    <element name=" Yuzolcumu " type="integer"/>
  </sequence>
</complexType>
(XML Schema)

```

```

<element name="Turkey " type="ex: TurkeyType"
substitutionGroup="gml:_Feature"/>
  <complexType name=" TurkeyType">
    <complexContent>
      <extension base="gml:AbstractFeatureType">
        <sequence>
          <element name= IL_ADI " type="string"/>
          <element name="PLAKA_NO " type="integer"/>
          <element name="NUFUS_1990" type=" integer " />
          <element name= NUFUS_1997" type=" integer" />
          <element name=" DEGISIM" type="integer"/>
          <element name=" ORT_GELIR_1980" type="integer"/>
          <element name=" Yuzolcumu " type="integer"/>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
(GML Schema)

```

An important benefit of GML is that it retrieves the best resolution image related to the geographic area. At the same time these maps which are translated to presentation format can be saved in the user's computer. Another benefit of GML is based on web technology. As a result, it can be visualized with web browsers on the user's computer without additional software necessity. In addition, it is different from raster maps. As a result, Content of Maps which are obtained from a GML document

are seen by opening the documents and reading them online. It is possible for GML documents to have different interpretation of GML documents according to different XSLT stylesheet files. It is known that XSLT stylesheet file translates GML documents to presentation language.

Update and insert operations can be done in an easy way on the geography Markup Language file which can be constructed with just a notepad application. Another property is that GML files can hold isolated both geographical–spatial information and non-spatial information related with spatial information in the same file. When visualized this information can be linked to related locations with the URL. And then it can be displayed with all information when clicking in the geographical information on the user’s computer. In addition, it is possible to filter operations in the geographical information. This operation is done by the querying of non-spatial information. Users have the opportunity to select themes for filtering results. These features are just as related to and worked on the GML file; it is not possible for other image formats. On the SVG documents which are obtained from GML documents, it is possible to work with JavaScript scripting language for adding animation to documents. Detail and objects which show the changes in time may be shown easily with animated graphics. (Komesli and Ünalır, 2004).

GML is a spatial data format. In GML documents, geographical information is coded and it is published by the OGC consortium which is a non-profit organization. It is a general format to translate data to each other. At the same time, GML is worked on new generation PDA and mobile phones which are xml based devices. The obtained benefit of using GML technology is that it is suitable for all mentioned usage as a unique format. (Komesli and Ünalır, 2004).

There are many properties in GML geographical markup language. It must be defined that all these properties in the schema document for getting a valid GML document. There is no special style information in a GML document. For this reason, if it is required to visualize GML documents on the web browser, it is necessary to translate the document to a presentation document which is suitable for display by a technology. This technology is XSLT technology which works on a web platform.

4.1 GML Schemas

It is possible to construct XPath usage which is used on the XSLT with help of schemas which change with GML version. For example, in GML Schema 2.1.2, two geometry elements “Point” and “Polygon” are as below. By looking at the these schema diagrams, detailed information which is related with GML geometry can be understood.

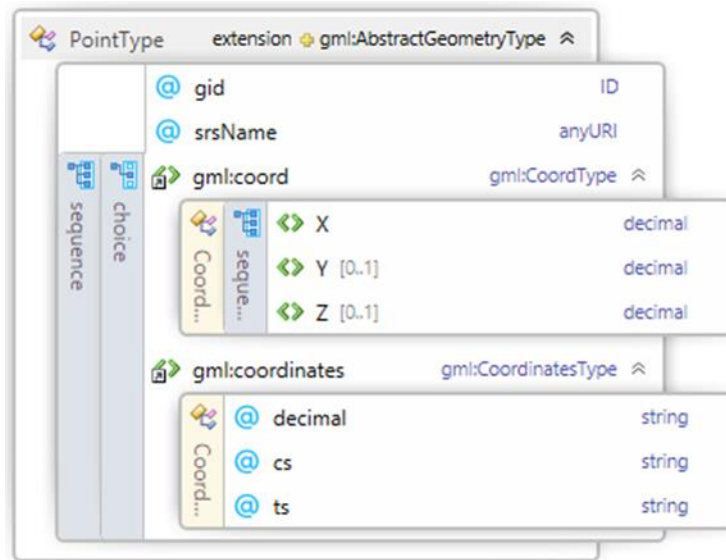


Figure 4.1.1 Geometry Elements of Point

In this way, as shown below, it is possible to handle all geometry in GML versions with writing various condition expressions for gml: Point elements in XSLT.

```
<xsl:if test="contains(/gml:coord/gml:X, ')"> . . . </xsl:if>  
<xsl:if test="contains(/gml:pos, ')"> . . . </xsl:if>  
<xsl:if test="contains(/gml:coordinates, ')"> . . . </xsl:if>  
<xsl:if test="contains(/gml:coordinates, ',)"> . . . </xsl:if>
```

In the same way, looking at the structure of the gml: Polygon It is possible to write the XSLT code associated with it.

```
<xsl:if
test="contains(//gml:Polygon/gml:outerBoundaryIs/gml:LinearRing/gml:coordinates,
')"> . . . </xsl:if>
```

```
<xsl:if test="contains(//gml:Polygon/gml:exterior//gml:pos,',')"> . . . </xsl:if>
```

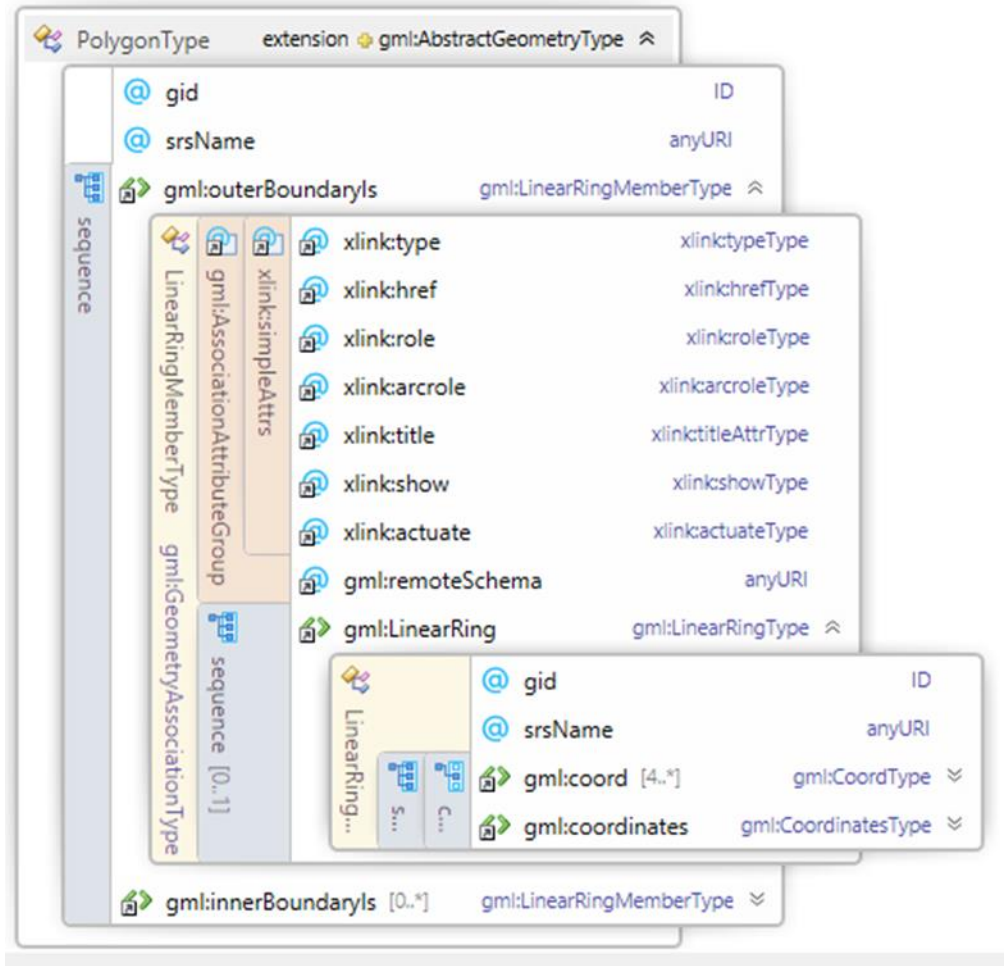


Figure 4.1.2 Geometry Elements of Polygon

```
<xsl:if test="contains(//gml:Polygon/gml:interior//gml:pos,',')"> . . . </xsl:if>
```

```
<xsl:if test="contains(//gml:Polygon/gml:exterior//gml:coordinates,' ')"> . . . </xsl:if>
```

```
<xsl:if test="contains(//gml:Polygon/gml:interior//gml:coordinates,' ')"> . . . </xsl:if>
```


5 Extensible Styleseet Language For Transformation (XSLT)

In this section, it is mentioned how XSLT (Extensible Stylesheet Translation Transformations) uses translation technology for visualization of spatial data documents which is produced with GML. XSLT, the extensible stylesheet language for transformations, is a language that is primarily designed for transforming one XML document into another (James Clark, 1999). XSLT is a language which provides the mechanism to transform and manipulate XML data. It is W3C recommendation (W3C-XSLT, 2014). The World Wide Web Consortium (W3C) started to develop XSL because there was a need for an XML-based Stylesheet Language. With XSLT you can add/remove elements and attributes to or from the output file. You can also rearrange and sort elements, perform tests and make decisions about which elements to hide and display, and a lot more. There is some mention of XSL (Extensible Stylesheet Language) language before mention of XSLT technology which is based on Document Style Semantics and Speciation Language (DSSSL) language. Because XSLT translation language is a small part of XSL (Extensible Stylesheet Language) extensible style document language, one wants to convert XML document which isn't a file format before it is defined. Thus, there is no information about XML document defined beforehand. So, it is not possible to decide to visualize which form of the document will result. For this reason, there is a need for an additional document which helps XML documents give information and to visualize them. This additional document is an XSL document. XSL documents have XSL specifications. These specifications are developed by the W3C consortium. Specifications are a language. This language defines what way should be defining for visualization of SVG document. XSL is a tool for producing of SVG graphics. It is a recommended technology for this process. There are three sections in XSL language structure. These are XSLT, XPath (XML Path Language) and XSL-FO (XSL Formatting Objects). XSLT translate GML documents to SVG documents. XPath addressing language search in GML documents. XLS-FO formats GML documents. XSLT Extensible Style Sheet language is constructed by uniting the XSLFO XSL technologies.

$$\text{XSL} = \text{XSLT} + \text{XSLFO} + \text{XPath}$$

XSL language is designed for adding style to XML documents with using Formatting Objects (FO). Objects which named as FO Objects is used for identify

objects in the page from XML. XSLT represents an important part of XSL Standards. XSLT translates documents which is XML based or not to other data documents. XSLT is pre-defined language translation technology. During the translation process, the target document is created without changing the source document. GML documents are the source document and SVG document is the target document on the translation of GML documents to SVG documents. As we mentioned before, we know that there is no information about meaning of the GML document inside. For this reason, it is needed to translate to a viewable format by translation technology of geographical/spatial data which is inside of the GML documents. In GML, data and presentation of data are separated from each other. With XSL technology, it is possible to translate of GML documents to other any formats. So, it is possible to presentation of GML documents. XSLT is a functional programming language and it is published by W3C (World Wide Web Consortium) Consortium which define valid style files. In XSLT style files, there is a set of templates which contain many of the rules. Thanks to these templates GML content can be showed as a map within a web page. XSLT files are created with an XML-based defined template language. One of the advantages of the working with XSLT translating technology is that it is possible to change just on the document. So, it is not necessary to compile the entire project. Another advantage is that it is suitable for different comments with different XSLT stylesheet files.

Like any other programming language, in XSLT translating language there are programming language components. These are “Data Types (Boolean, number, external type definitions)”, “Operations(<xsl:template>, <xsl:apply-templates>, <xsl:match>, <xsl:sort>, <xsl:output> etc.)”, ” Flow control mechanisms(<xsl:if>, <xsl:for-each>, <xsl:choose> etc.)” and other programming language structures definitions.

The first noticeable advantage of the XSLT is that it is an Object-oriented language. It is the platform independent. Changes are made as independent from projects. It has an interface which is standardized for positional operations.

XSLT documents should be prepared as well-form and complete form for translating process. At the same time, it should implement specification rules. First of all, XSLT passes the document and a tree structure is created. It starts the root element of the constructed tree structure and scans the tree and then translates the

document according to the rules defined in the XSL style sheet. XSL style sheet consists set of one or more rules and it contains rules which are applied to the matched node. Rule set in XSL document is named as a template. To construct templates, <xsl:template> elements are used. XSLT translates every one of the XML elements to SVG elements. When XSLT is scanning on the tree, it uses XPath expressions. So XSLT accesses the different parts of the tree (Kılınç, 2014). The structure of the XSLT document is like the structure of an XML document. Both of them start with <?xml version="1.0" encoding=" utf-8"?> element. And they continue with <xsl:stylesheet> or <xsl:transform> root element. Specifications which are defined in root elements obtain validation of XSLT style document. For reaching element and property of XSLT, XSLT namespace and version number are used. Namespace of W3C XSLT is “xmlns:xsl=<http://www.w3.org/1999/XSL/Transform>”. It is known that specifications in the <xsl:stylesheet> element are identified. And also the version and encoding properties within it are identified. Other elements that are located in the root element are as child elements. Below there is an example of the explained structure.

```
<? Xml version="1.0" encoding="ISO-8859-1">
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
</xsl:stylesheet>
```

The following element is the <xsl:template> element. There is an SVG element in the <xsl:template> for creating a map of which the format is SVG. <xsl:template match="/"> defines the template which is applied to the root of the document. This element is closed with </xsl:template>. The template which is located at every XSLT translation documents contains “match” property. With this command, the desired elements which are in the source are obtained. ”match” associates the template with the XML element. In addition, it is possible that a unique template is defined for all XML files. The value of the “match” property is “XPath” path definition. It is accessed via the element located in the source documents by “XPath”. References of the elements are introduced according to xPath rules.

XPath is a language. This language is used for referencing of selected special section of GML documents and getting around according to these references. It is like SQL querying language. XPath helps to find data, in other words, it helps to reach

elements and attribute values in desired characteristics in the documents. XPath makes necessary definition for selecting the nodes to be included into a loop. We know that XPath can create nodes lists. It can also produce new values with using a set of functions that help. Also, it can make comparisons. It can be identified every nodes of path in XPath and XML documents. During this process, it can be got nodes list by using a querying operation. In this language, syntax is used which is like an extensible text-based path/file. Path/file syntax is a path which is obtained by addressing of the tree's special sections. In other words, XPath is a assistive technology for defining a pattern. Some operators are needed to to make a transaction with the XPath. These operators are addition (+), subtraction (-), division (div), multiplication (*), Equality Test (=), Inequality Test (!=) , The Smallness Test (<), Small Equality Test (<=), Size Test(>), Large Equality Test (>=), logical or (or), logical and (and) and Parts of the remaining (mod). <xsl:foreach> element which is structure of XSL language helps to reach elements in the loop. Reached elements are in the XML document. <xsl:value-of > is another structure of the XSL language. This element helps to select the value of the specified element. In the example below, there is a template which is prepared on XSL for "gml:Polygon" element which is located in the GML document.

```

<xsl:template match="gml:Polygon">
  <xsl:for-each
select="//gml:Polygon/gml:exterior//gml:coordinates">
  <xsl:variable name="clist">
    <xsl:value-of
select="//gml:Polygon/gml:exterior//gml:coordinates"/>
  </xsl:variable>
<polyline points="{ $clist}" style="fill:lime;stroke:purple;stroke-
width:5;fill-rule:evenodd;"/>
  </xsl:for-each>
</xsl:template>

```

In the example, "polygon" element in GML documents is translated to "Polyline" element in SVG document. Coordinate values in the "Polygon" element are reflected to the "Polyline" element. With this template, it is possible to handle all "Polygon" elements in the GML document as a "Polyline" element in the SVG document. In other words the template is applied to the GML document and then the SVG elements

are produced. Like this template, there are many templates for translating the GML elements to SVG elements.

A loop is constructed for gml:coordinate elements with `<xsl:for-each select="//gml:Polygon/gml:exterior//gml:coordinates">`. An example of this XPath is `select="//gml:Polygon/gml:exterior//gml:coordinates`. With this XPath expression it is possible to reach desired section of tree. By using `<xsl:value-of select="//gml:Polygon/gml:exterior//gml:coordinates"/>` the value of the “coordinates” element which is element of “polygon” element is obtained. In the `gml:exterior//gml:coordinates` expression there is “//” expression. With this expression, it is possible that two elements of the expression can be written without specification. Like this expression, there are various kinds of expressions which are working with elaboration. These elaborations are displayed below.

Expression	The Meaning Of
/	/kitaplar expression defines <kitaplar> element. /kitaplar/kitap expression selects <kitap> element which is child element of <kitaplar> element.
//	It selects all element in the document regardless of where you are. The selection start to match the current from the node.
@	It selects property of a node. The experssion of /kitaplar/kitap/@id is selects id named property of the <kitap> element.
*	It selects all child elements in path identification. /kitaplar/kitap/* expression selects all child elements of <kitap> elements.
.	It shows current node.
..	It shows the root node. If current node is <ad>, “..” expression shows <kitap> node.
[]	It defines a selection criterion. /kitaplar/kitap[yazar=“Mevlana“] expression selects elements which matches the criteria and it contains <yazar> element.
starts-with	It selects elements according to initial characters of the text of an element. /kitaplar/kitap[starts-with(yazar, 'R')] expression finds all <kitap> elements which starts with “R” letter and which contains text and which has <yazar> element
position	It selects elements by looking at the location.

	/kitaplar/kitap[position()=2] selects second <kitap> element.
count	It calculates count of the specified node. count(kitap) expression retrieves count of <kitap> elements

Table 5.1 XPath Expressions

Below, there are some XSLT codes of this thesis. It can be seen that calculated “height” and “width” values which are located in the GML document with this XPath codes. These calculated values are transferred to the parameters. These values are parameter values which are used for the constructed SVG file. With this example, it can be seen that XSL is a programming language.

```

<xsl:param name="minY" select="substring-after(//gml:boundedBy/gml:Envelope/gml:lowerCorner,
)"/>
<xsl:param name="minX" select="substring-
before(//gml:boundedBy/gml:Envelope/gml:lowerCorner,')"/>

<xsl:param name="maxY" select="substring-
after(//gml:boundedBy/gml:Envelope/gml:upperCorner,')"/>
<xsl:param name="maxX" select="substring-
before(//gml:boundedBy/gml:Envelope/gml:upperCorner,')"/>

<xsl:param name="height" select="$maxY - $minY"/>
<xsl:param name="width" select="$maxX - $minX"/>

<xsl:template match="/">
<svg width="100%" height="768" viewBox="$minX $minY 1024 768"
xmlns="http://www.w3.org/2000/svg" xml:space="preserve" onload="init(evt)" style="shape-
rendering:geometricPrecision; text-rendering:geometricPrecision; image-rendering:optimizeQuality;
fill-rule:evenodd; clip-rule:evenodd; stroke:black; stroke-width:1; fill:none;". "All SVG codes are
here" </svg>
</xsl:template >

```

The scope of the thesis, if it is mentioned to code blocks of XSLT section of the thesis, there is “North” indicator. SVG codes of this indicator like that below. With this code block, it is possible to visualize this SVG objects like at the picture in Figure5.1.



Figure5.1 North Arrow with SVG Format

```

<g id="north_arrow">
<path          d="M13.652,42.749v-7.164h1.412.943,4.807v-4.807h1.336v7.164h-1.4431-2.9-
4.705v4.705H13.652z"/>
<circle fill="none" stroke="#000000" stroke-width="1" cx="16.417" cy="39.167" r="6.417"/>
<polyline    fill="none"    stroke="#000000"    stroke-width="1"    stroke-miterlimit="10"
points="16.417,32.75 16.417,4.017 21.625,18.157 16.458,18.157"/>

</g>

```

In addition, there is an XSLT code block which translates GML LineString elements to the SVG polyline or path elements.

```

<xsl:template match="gml:LineString">
  <xsl:for-each select="//gml:LineString">
    <xsl:variable name="linearRingCoord12">
      <xsl:value-of select="./gml:coordinates"/>
    </xsl:variable>
    <xsl:variable name="ID">
      <xsl:value-of select="@gml:id" />
    </xsl:variable>
    <polyline id="{ $ID }" class="path" points="{ $linearRingCoord12 }"/>
    <path id="{ $ID }" class="polyline" d="{ $linearRingCoord12 }" >

      <set attributeName="opacity" from="1" to="0.5" begin="mouseover" end="mouseout"/>
      <set  attributeName="fill"    begin="mouseover"    end="mouseout"    from="#000099"
to="#CCCCFF"/>

    </path>

  </xsl:for-each>
</xsl:template>

```

In the above example there is the **<xsl:template match="gml:LineString">** expression for catching all gml:LineString elements which is in the GML document. All processing instructions are defined for these elements in this code block. With the code block of **<xsl:variable name="linearRingCoord12"><xsl:value-of**

select="/gml:coordinates"/> </xsl:variable>, it is possible to access to gml:coordinates values which is in the gml:Linestring element.

5.1 XSL-FO XSL Formatting Objects

XSL-FO XSL formatting objects are objects which are used for formatting the XML files. The formatting process is a process that converts to a format that is understandable by the user appropriate to the XSL translation results. It is required that XSLT documents contain SVG elements which are desired to be located in the SVG document. In other words, SVG elements are created as descriptive in the XSLT template document. There are XSLT processors which make the conversion process within the XSLT document prepared by program. These processors take parameters. The input parameter is the GML document. The mentioned GML document is a data format which intended to be translated to the SVG data format. Input and output parameters are basic structures that determine the structure of an XSLT style sheet files. In Figure 5.1.1, there is an example of the working mechanism.

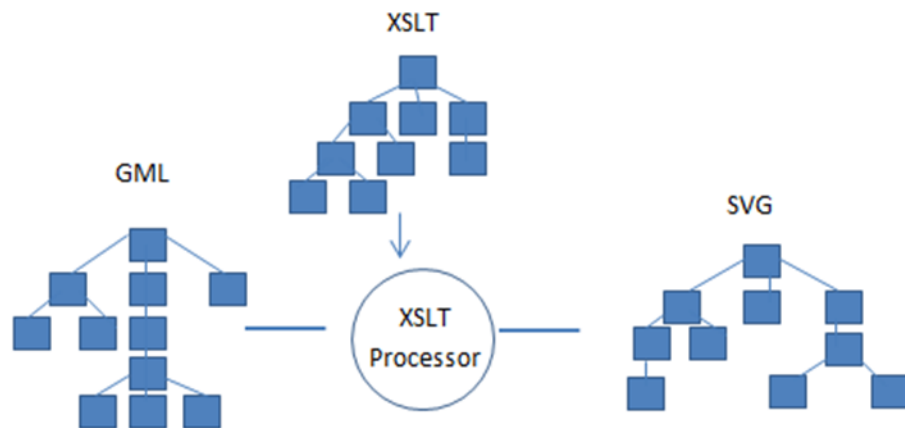


Figure 5.1.1 XSLT Processor Functionality

The working mechanism of the XSLT processors is seen in the Figure 5.1.1. According to the figure, all components are in a tree structure. GML, SVG and XSLT tree structures are work together. XSLT processor translates GML tree structure to SVG tree structure. During the translation processes, the XSLT tree structure works actively. In the processors, the pairing process is executed. The pairing process is

defined in the XSLT templates which would be among those which are appropriate to the GML file. If pairing is successful, SVG codes which are located in the template are included in the output document. In Figure 5.1.2, there is a showcase of the process.

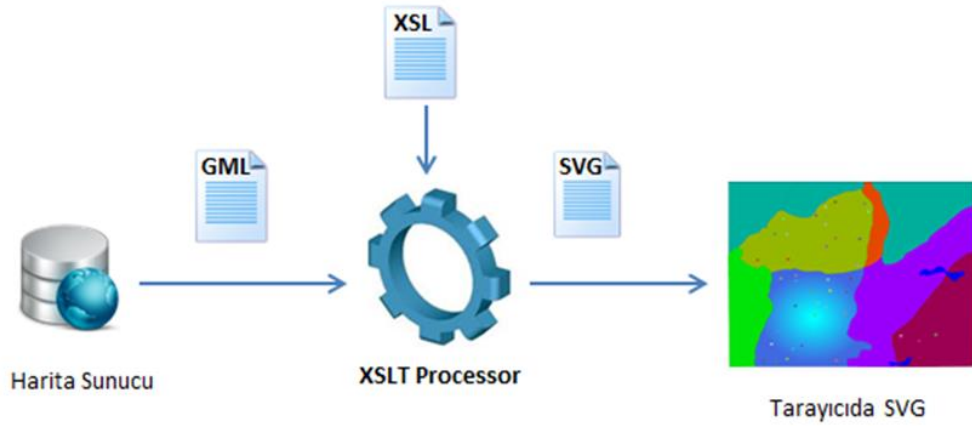


Figure 5.1.2 Translating of GML to SVG

6 Scalable Vector Graphics (SVG)

The most basic goal of GML documents is that it provides and transfers the geographical data contents. But for presentation process, it must be converted to an appropriate language for the presentation format with the help of a tool. The presentation form of a map to the user is an important issue. It must be an understandable interface of an appropriate map. In addition, it should be in a high resolution and functional. There three graphic processing language formats. These are SVG (Scalable Vector Graphics), VML (Vector Markup Language) and Web 3D. Of these languages, SVG is a language that is designed for interoperability with other applications with the W3C ' specifications. In other words, SVG vector graphics language is a markup language published by W3C consortium and it is used for creating two-dimensional graphics images (SVG 1.1 Specification, 2003). One of the reasons for it being the most preferred technology is bandwidth size for visualization. In this case, SVG vector graphics creating format is the first order for maps which is obtained from positional information. Because of the need for minimum bandwidth size, it downloads to the user computer rapidly. In additional, map objects/vectors are reachable during the searching process.

The vector elements can be given interactivity property with client side scripting language (JavaScript, jQuery). The vector elements are both XML-based and a web technology. So, these elements support all kinds of vector elements. For this reason, SVG vector graphics are both interactive and dynamic. Another advantage is that it is possible to work with other web-based and XML based technologies. With these technologies there is a very good harmony. In the scope of this thesis, SVG works with some technologies. These technologies are XSLT, GML, CSS web based and XML based technologies. It is important to know high-level programming about these technologies to develop this thesis. It is possible to maintain the SVG content as separate files in .svg format. And then it is possible to include a web page which will be used with the help of various labels.

There are three types of graphical objects in SVG vector graphics. These graphical objects are “Vector Graphic Shapes”, “Raster Graphic Shapes” and “Text” types. Style can be added for all these types with the help of the CSS. Applied styles support to the developer. With user-side scripting language there is lot of

functionality added to map. The topic of “Styling with CSS” is handled in following section. SVG objects can be added to a web page like any other tags.

6.1 Structure of SVG Vector Graphics

It is possible to display vector graphics as different document formats like XML and HTML. SVG documents of the definitions required for the display as follows.

```
<?xml version="1.0"?>
```

```
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.0//EN" "http://www.w3.org/TR/2001/REC-SVG-20010904/DTD/svg10.dtd">
```

The start tag of the SVG document is `<svg>` tag. Within this element, many properties can be defined which belong to the SVG document. To give an example, there are width and height values in the SVG document. In addition, except for the width and height of the SVG document, there is an attribute named as “viewbox”. “viewbox” is visibility setting in the document. “viewbox” determines the boundaries of the visibility rectangle. Below, there is an example of SVG root element which is defined with all of the features and related definition about the element.

```
<svg width="20cm" height="10cm" viewBox="0 0 8000 300" fill-opacity="1" color-rendering="auto" color-interpolation="auto" textrendering="auto" stroke="black" stroke-linecap="square" stroke-miterlimit="10" shape-rendering="auto" strokeopacity="1" fill="black" stroke-dasharray="none" fontweight="normal" stroke-width="1" xmlns="http://www.w3.org/2000/svg" fontfamily="sansserif" font-style="normal" strokelinejoin="miter" font-size="12" stroke-dashoffset="0" imagerendering="auto">
...
</svg>
```

6.1.1 Circle Drawing

Circle is used for generally defining real-world objects, such as a building, city, and stop. Circle has three parameters. These parameters are `cx`, `cy` and `r` properties. “`cx`” and “`cy`” specify the center point of the circle in coordinate system. “`r`” is radius

feature. Below there is an example which is related with drawn circle in a SVG document.

```
<svg xmlns=http://www.w3.org/2000/svg height="100" width="250">  
<circle cx="100" cy="60" r="35" stroke="blue" stroke-width="5" />  
</svg>
```



Figure 6.1.1.1 Circle Drawing with SVG

6.1.2 Polyline Drawing

LineString element in GML documents is translated to Polyline elements in SVG format. Polyline has “points” attributed and this attribute takes values as (x, y) coordinates pairs. Thus, a point attribute is shown like the following expression `points="x1, y1, x2, y2..."`. At the below there is an example which is related with drawn Polyline in a SVG document.

```
<svg xmlns=http://www.w3.org/2000/svg height="100" width="250">  
<polyline points="0,50 70,50 70,120 140,120 140,50 210,50" style="fill:white.stroke:blue:stroke-width2;" > </polyline>  
</svg>
```

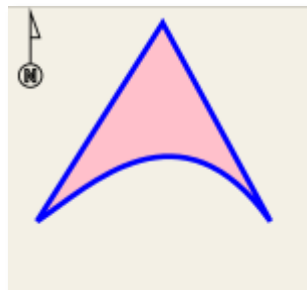


Figure 6.1.2.1 Polyline Drawing with SVG

6.1.3 Polygon Drawing

Polygon element in GML documents is translated to Polygon elements in SVG format. Polygon has “points” attributed and this attribute takes values as (x, y) coordinates pairs. Point attribute is shown like the following expression points=“x1, y1, x2, y2...”. Unlike the Polyline element, Polygon element closes the drawing path. [8] (Çelikbilek, 2011)

```
<svg xmlns=http://www.w3.org/2000/svg height="100" width="250">  
<polygon points="110,50 160,70 166,122 136,163 82,163 50,122 110,110" fill="blue" stroke="pink"  
stroke-width="10" /></svg>
```

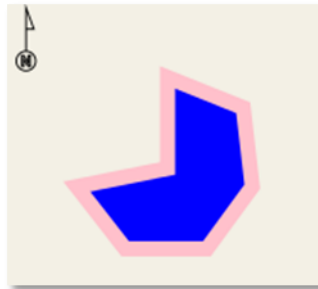


Figure 6.1.3.1 Polygon Drawing with SVG

6.1.4 Path Drawing

One of the elements that satisfy LineString elements in GML documents is path element. It is used for drawing straight lines as well as curved lines as a path.

```
<svg xmlns=http://www.w3.org/2000/svg height="100" width="250">  
<path d="M 20 120 L 90 10 150 120 C 100 50 50 100 20 120" fill="pink" stroke="blue" stroke-  
width="3" />  
</svg>
```

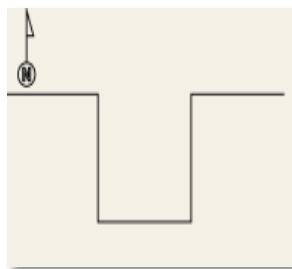


Figure 6.1.4.1 Path Drawing with SVG

Path element contains “d” attribute for identifying point coordinates instead of “points” attribute. The path information is created with commands and it draws a path segment according to the coordinate information which is typed after each command sequence. [8] (Çelikbilek, 2011). The following commands can be used for this element of attribute.

M (moveto): Coordinates **which are written after moveto commands** creates the starting point of the drawing. M command don't draw a path, it makes definition. It is used in the form of M (X, Y) +.

L (lineto): It is used in the form of L (X, Y) +. It is drawn lines between the given coordinate values after L. In addition, naturally, it can be specified in more than one coordinate pair after L.

H (horizontal lineto): It is used in the form of H (x) +.

V (Vertical lineto): It is used in the form of V(y)+.

C (curveto): It is used in the form of (x1 y1 x2 y2 x y)+. (x, y) is the endpoint. (x1, y1) ve (x2,y2) are control points.

S (smooth curveto): It is used in the form of (x2 y2 x1 y1)+ .

Q (quadratic belzier curve): It is used in the form of (x1 y1 x y)+.

T(smooth quadratic Belzier curveto): It is used in the form of (x, y)+.

A (elliptical Arc): It is used in the form of (rx ry x-axis-rotation large-arc-flag sweep-flag x y)+.

Z (closepath): This command sets the path drawing to turn off. It does not receive the next coordinates.

Below, there is an example which is used this path coordinates in GML code. These codes are translated into SVG codes by XSLT.

```
<gml:LineString gml:id="İzmir">
  <gml:coordinates>
    M 82.4375,215.0625 L 80.90625,215.84375 L 80.875,215.84375 L 80.8125,215.875 L
    77.625,216.71875 L 74.78125,218.71875 L 72.5,220.21875 L 72.375,220.3125 L 72.25,220.3125 L
    70.15625,220.4375 ...
  </gml:coordinates> </gml:LineString>
```

In the following example, there is an XSLT code which translates all the elements of a "Point" to the SVG element. In here, "Point" object which is element of the GML document is translated to "Circle" object which is element of the SVG document. As above, coordinate values are taken into the parameters and transferred in accordance with the programming logic.

```

<xsl:template match="gml:Point">
  <xsl:if test="contains(/gml:pos,' ')">
    <xsl:variable name="xCoord"> <xsl:value-of select="substring-before(/gml:pos,' ')"/>
  </xsl:variable>
    <xsl:variable name="yCoord"><xsl:value-of select="substring-after(/gml:pos,' ')"/>
  </xsl:variable>
    <g transform="translate(204.928128,91.700833)">
      <g transform="rotate(180)">
        <circle cx="{xCoord}" cy="{yCoord}" r="6.417" stroke="black" stroke-width="1"
fill="pink"/>
      </g>
    </g>
  </xsl:if>
  ...
</xsl:template>

```

It is seen above that commands are aimed to obtain vector representations of "Point" elements which are GML elements. So, these elements is translated to "Circle" elements in SVG. There are a lot of elements translated from GML to SVG. These are like this: "Box" element in GML is translated to "rect" element in SVG. "Point" element in GML can be translated to "rect", "circle" or "path" elements in SVG. "LineString" element in GML document is translated to "polyline" or "path" element in SVG. "Polygon" element in GML is translated to "path" or "polyline" elements in SVG.

6.2 GML Documents to SVG Graphics

Geography Markup Language (GML) documents are visualized as Scalable Vector Graphics (SVG) graphics on the web browser with Extensible Stylesheet Language Transformations (XSLT) transformation technology. Produced SVG document likes GML document as structure. As we see below, there are a GML documents and produced SVG documents from this document. The translation can be programmed manually or with a transformation engine. As is seen below, we can notice that there are pre-defined style files, StyleSheet1.css, on the constructed SVG elements. These style file handles defined styles of produced SVG elements and the elements are gained style properties by style file.



<pre> <?xml version="1.0" encoding="UTF-16"?> <?xml-stylesheet href="StyleSheet1.css" type="text/css"?> . . . <gml:Polygon class="CHANIA"> <gml:exterior> <gml:LinearRing> <gml:coordinates>1327863,- 1799582 1327957,-1799520 1327731,-1799043 1327403,- 1798753 1326983,-1798467 1326792,-1798758 1326250,- 1798439 1326412,-1798832 1327148,-1799166 ... </gml:coordinates> </gml:LinearRing> </gml:exterior> </gml:Polygon> <gml:Polygon class="RETHYMNI"> <gml:exterior> <gml:LinearRing> <gml:coordinates>1398258,- 1817477 1398448,-1818461 1399298,-1820141 1400238,- 1821240 1400642,-1822566 1400689,-1822688 1400964,- 1823927 1400848,-1824297 1399553,-1825172 ... </gml:coordinates> </gml:LinearRing> </gml:exterior> </gml:Polygon> </pre>	<pre> <?xml version="1.0" encoding="UTF-16"?> <?xml-stylesheet href="StyleSheet1.css" type="text/css"?> <!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.0//EN" "http://www.w3.org/TR/2001/REC- SVG-20010904/DTD/svg10.dtd"> <svg width="100%" height="100%" viewBox="1290000 1750000 300000 120000"> <title> Poligon Geometrasi </title> <g transform="scale(1,-1)" id="poligonlar"> <polygon class="CHANIA" points="1327863,-1799582 1327957,-1799520 1327731,-1799043 1327403,-1798753 1326983,-1798467 1326792,-1798758 1326250,-1798439 1326412,-1798832 1327148,-1799166 ... "/> <polygon class="RETHYMNI" points="1398258,-1817477 1398448,-1818461 1399298,-1820141 1400238,-1821240 1400642,-1822566 1400689,-1822688 1400964,-1823927 1400848,-1824297 1399553,-1825172 ... "/> <polygon class="IRAKLEIO" points="1479422,-1792300 1479454,-1792012 1479145,-1791370 1479293,-1790409 1479089,-1789682 1479180,-1789566 1479525,-1789655 1479702,-1789229 1479055,-1789361 ... "/> </pre>
---	--

7 CSS Styling Language

Cascading Style Sheet (CSS) styling technology proposed for World Wide Web Consortium (W3C) in 1996, acts as major role to style and visualize information on the web browser of web technology (HTML, XML, GML etc.) which has label mechanism. Many visualizing settings like document color, font properties, spacing settings can be done by CSS style definitions. CSS styling mechanism which is supported from many web browsers first scans all nodes from start to end of the document tree and then makes implementation of the style to available nodes.

Using GML data with external CSS-style sheet files, it is possible to arrange SVG graphic files only over style sheet externally. Like styling mechanism of HTML technology with CSS style files, it is possible to style SVG files with CSS technology. It is known that CSS style files are defined as externally during the visualization of the GML. One reason to prefer the SVG technology is that graphics are independent from resolution, drawing processes are done from pre-defined SVG labels and graphical elements can be reachable as one of the elements of Document Object Model (DOM) architecture. Another method for Mapping of geography data is pixel based <canvas> element which is constructed from Html5 for image processing. With <canvas> element, it is possible that construct high quality raster maps, at the same time, it is possible to define style definitions for canvas elements. It should be noted that SVG and Canvas technics are used differently from each other. Canvas elements may be preferred when interactive and pixel based image editing, image analyzing, other hand SVG may be preferred when constructing user interface which are independent from resolution. (Çelikbilek, 2011).

A CSS is structured in 3 steps. The first step, defining a “selector” for noted which nodes are selected. After that, defining a “property” definition which corresponds to CSS syntax and finally, giving a value to defined property.

7.1 Basic CSS Selectors

We can group CSS selectors for reaching to CSS objects by three categories. These are ID Selectors, Class selectors and Label Selectors. In addition to these selectors there are ordinary selectors. Below, the ordinary selectors are listed. When grouping CSS selectors it is considered that how reach their objects.

Child Selectors (E>F): It is used for selecting all F elements inside the E sub-hierarchy.

Neighbor Sibling Selectors (E + F): It is used for selecting E and F elements which are located on same hierarchy layer.

Descendant Selectors (E F): It is used for selecting F elements which are located on under of E element.

Global Selectors (*): A character which figures all elements on the document,*, is a selector for describing all features which is wanted to valid for all elements.

Attribute Selectors E [attribute= value]): It is used for selecting elements which have defined properties (href, alt, title etc.) and have pointed values.

It is possible that keep style definition rules and CSS files minimum and understandable during selecting operation. One of the rules that when the same properties needs to apply to DOM elements which have element's properties (Id, Class, Attribute etc.) different from each other, grouping is separated by comma.

```
.class1, #id1, #id2, .class2 { /* styles which are valid for all elements*/ }
```

A rule of classes which are only located on a definite location of document is selecting class elements pointed at special area with using the structure "LabelArea. ClassName". At the below, it is showed that defined style rules for "Bölgeler" and "Ülkeler".

```
Polygon. Ülkeler {fill-opacity: 0.7; stroke-width: 0.2; label-anchor: 0.5 0.5;}
```

Although CSS files can be constructed by only using Notepad application, Defining process can be increased by using advanced CSS editors. Some of these editors are CSS SPY 1.0 RC 1, RAPID CSS 2010, TOPSSTYLE 4 and STYLE MASTER 4.6.

7.2 CSS Usage in SVG Files

Stylizing with Cascading Style Sheet (CSS) is a unique method for external styling graphic elements which are produced from translating of Geography Markup Language (GML) elements to SVG elements. A style provided SVG graphic reflects style properties in the browser. CSS which is independent from XML (Extensible Markup Language) and XSLT (Extensible Style sheet Language Transformations) avoids being dependent on a unique style SVG data which is produced from GML. There are various approaches for implementing CSS files to a SVG files. The most preferable method is the external definition method, method 1, due to easy administration. These are listed at the below.

Method1. With definition of external style sheet, it can be given style properties to polygon vector like shown at the example. In the “StyleSheet1.css” file, it is identified styles at the below and then the file is included to SVG document.

Step 1: polygon {fill: darkseagreen; stroke: #000000; stroke-width: 0.37px;}

Step 2: “StyleSheet1.css” is introduced to SVG documents. This process is like in the following example.

```
<?xml version="1.0" encoding="UTF-16"?>
<?xml-stylesheet href="StyleSheet1.css" type="text/css"?>
<svg version="1.1" id="Layer_1" xmlns="http://www.w3.org/2000/svg" width="599.884px"
height="379.041px" viewBox="0 0 599.884 379.041" xml:space="preserve">
<g> <polygon points="56.956,81.045 102.162,86.689 107.813, 89.915 111.849,88.303
114.271,88.303 116.693,90.722 115.886,93.14 117.501,95.56 122.343, 96.365 123.152,99.591
124.766, 102.01 128.803, 102.816 132.838,101.204 140.103,101.204 143.333, 97.171 147.369,
95.56 147.369,100.398 140.91, 104.428 138.489,104.428 136.875,108.46 137.682, 109.267
136.875,110.878 129.61,111.685 128.803, 114.104 124.766,119.749 123.958, 116.523
123.958,115.717 121.536,120.555 121.536,126.199 111.849, 133.456 110.235, 133.456
107.006,140.712 107.006, 147.969 106.198,152 104.583,153.614 102.969,153.614 101.354,
141.519 94.897,139.906 91.667, 139.101 90.86,143.132 85.209,139.906 79.559, 139.906
71.486, 150.388 69.064,148.775 68.258,139.906 65.834,139.101 64.22,140.712 62.606,
139.906 60.184,133.456 56.956,132.65 56.148, 133.456 50.497,133.456 45.653,129.423
40.81,129.423 40.81, 127.005 36.774,122.167 36.774, 114.911 39.195,114.104 38.389,111.685
38.389, 106.042 50.497, 84.271 52.111,84.271 53.726,86.689 "/></g></svg>
```

Method 2: As a different approach, it can be defined style internally with <style> element in the upper section of related document. In order to perform this operation, <style> element is constructed internally upper of the document shown in the following example.

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE      svg      PUBLIC      "-//W3C//DTD      SVG      1.1//EN"
"http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">
<svg version="1.1 xmlns="http://www.w3.org/2000/svg" width="599.884px" height="379.041px"
viewBox="0 0 599.884 379.041" xml:space="preserve">
<defs><style type="text/css">
#poligonözelikleri{ fill:coral; stroke:#000000; stroke-width:0.50px; stroke-miterlimit:10;}
</style></defs>
<g>
<polygon id=" poligonözelikleri " points="56.956,81.045 102.162,86.689 107.813, 89.915
111.849,88.303 114.271,88.303 116.693, 90.722 115.886,93.14 117.501,95.56 122.343,
... ">
</g></svg>
```

Method 3: It can be styled with inline CSS definition. As an example of the method as below.

```
<g> <polygon style="fill: darkturquoise; stroke: greenyellow; stroke-width:2px; stroke-miterlimit:10;"
points="56.956,81.045 102.162,86.689 107.813, 89.915 111.849,88.303 114.271,88.303 116.693,
90.722 115.886,93.14 117.501,95.56 122.343, ... ">
</g>
```

For example there is below the “Turkey” map which is styled with CSS. These styles are listed below.

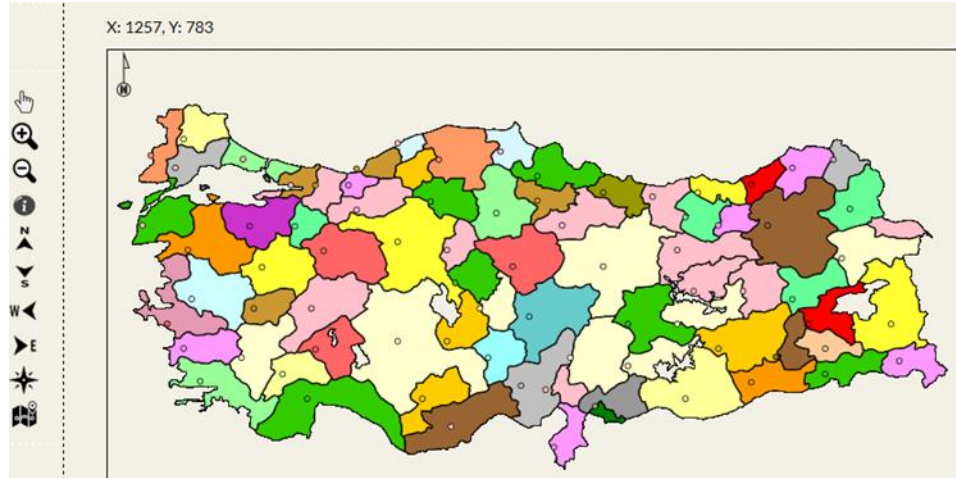


Figure 7.2.1. 7.2 CSS Usage in SVG Files

```
#Kırklareli{fill:rgb(255, 255, 153);stroke:black;stroke-width:1; }
```

```
#Edirne{fill:rgb(255, 153, 102);stroke:black;stroke-width:1; }
```

```
#Kastamonu{fill:rgb(255, 153, 102);stroke:black;stroke-width:1; }
```

```
#Sinop{fill:fill:rgb(255, 255, 153);stroke:black;stroke-width:1; }
```

```
#Bartın{fill:fill:rgb(255, 255, 153);stroke:black;stroke-width:1; }
```

```
#İstanbulAvr{fill:rgb(153, 255, 153);stroke:black;stroke-width:1; }
```

```
#Tekirdağ{fill:rgb(192, 192, 192);stroke:black;stroke-width:1; }
```

```
#Ardahan{fill:rgb(192, 192, 192);stroke:black;stroke-width:1; }
```

```
#Samsun{fill:rgb(51, 204, 0);stroke:black;stroke-width:1; }
```

```
#Artvin{fill:rgb(255, 153, 255);stroke:black;stroke-width:1; }
```

```
#Zonguldak{fill:rgb(204, 153, 51);stroke:black;stroke-width:1; }
```

```
#Karabük{fill:rgb(255, 204, 0);stroke:black;stroke-width:1; }
```

```
#Rize{fill:rgb(255, 0, 0);stroke:black;stroke-width:1; }  
  
#İstanbulAnd{fill:rgb(153, 255, 153);stroke:black;stroke-width:1; }  
  
#Kars{fill:rgb(102, 255, 153);stroke:black;stroke-width:1; }  
  
#Kocaeli{fill:rgb(204, 153, 51);stroke:black;stroke-width:1; }  
  
#Çorum{fill:rgb(153, 255, 153);stroke:black;stroke-width:1; }  
  
#Sakarya{fill:pink;stroke:black;stroke-width:1; }  
  
#Düzce{fill:rgb(255, 153, 255);stroke:black;stroke-width:1; }  
  
#Ordu{fill:rgb(153, 153, 0);stroke:black;stroke-width:1; }  
  
#Trabzon{fill:rgb(255, 255, 51);stroke:black;stroke-width:1; }  
  
#Erzurum{fill:rgb(153, 102, 51);stroke:black;stroke-width:1; }  
  
#Çankırı{fill:rgb(51, 204, 0);stroke:black;stroke-width:1; }  
  
#Bolu{fill:pink;stroke:black;stroke-width:1; }  
  
#Amasya{fill:rgb(204, 153, 51);stroke:black;stroke-width:1; }  
  
#Çanakkale1 {fill:rgb(51, 204, 0);stroke:black;stroke-width:1; }  
  
#Giresun{fill:pink;stroke:black;stroke-width:1; }  
  
#Tokat{fill:pink;stroke:black;stroke-width:1; }  
  
#Yalova{fill:pink;stroke:black;stroke-width:1; }  
  
#Gümüşhane{fill:rgb(102, 255, 153);stroke:black;stroke-width:1; }
```

```
#Balıkesir{fill:rgb(255, 153, 0);stroke:black;stroke-width:1; }  
  
#Bursa{fill:rgb(204, 51, 204);stroke:black;stroke-width:1; }  
  
#Çanakkale2 {fill:rgb(51, 204, 0);stroke:black;stroke-width:1; }  
  
#Ankara{fill:rgb(255, 255, 51);stroke:black;stroke-width:1; }  
  
#Bilecik{fill:rgb(102, 255, 153);stroke:black;stroke-width:1; }  
  
#Çanakkale3 {fill:rgb(51, 204, 0);stroke:black;stroke-width:1; }  
  
#Bayburt{fill:rgb(255, 153, 255);stroke:black;stroke-width:1; }  
  
#Sivas{fill:rgb(255, 255, 204);stroke:black;stroke-width:1; }  
  
#İğdır{fill:pink;stroke:black;stroke-width:1; }  
  
#Kırıkkale {fill:pink;stroke:black;stroke-width:1; }  
  
#Eskişehir{fill:rgb(255, 102, 102);stroke:black;stroke-width:1; }  
  
#Ağrı{fill:rgb(255, 255, 204);stroke:black;stroke-width:1; }  
  
#Yozgat{fill:rgb(255, 102, 102);stroke:black;stroke-width:1; }  
  
#Erzincan{fill:pink;stroke:black;stroke-width:1; }  
  
#Kütahya {fill:rgb(255, 255, 51);stroke:black;stroke-width:1; }  
  
#Kırşehir {fill:rgb(51, 204, 0);stroke:black;stroke-width:1; }  
  
#İzmir{fill:rgb(255, 102, 102);stroke:black;stroke-width:1; }  
  
#Manisa{fill:rgb(204, 255, 255);stroke:black;stroke-width:1; }
```

```
#Tunceli{fill:pink;stroke:black;stroke-width:1; }  
  
#Van{fill:rgb(255, 255, 51);stroke:black;stroke-width:1; }  
  
#Muş{fill:rgb(102, 255, 153);stroke:black;stroke-width:1; }  
  
#Bingöl{fill:pink;stroke:black;stroke-width:1; }  
  
#Nevşehir{fill:rgb(255, 255, 204);stroke:black;stroke-width:1; }  
  
#Afyon{fill:pink;stroke:black;stroke-width:1; }  
  
#Kayseri{fill:rgb(102, 204, 204);stroke:black;stroke-width:1; }  
  
#Konya{fill:rgb(255, 255, 204);stroke:black;stroke-width:1; }  
  
#Elazığ{fill:rgb(255, 255, 204);stroke:black;stroke-width:1; }  
  
#Bitlis{fill:rgb(255, 0, 0);stroke:black;stroke-width:1; }  
  
#Malatya{fill:rgb(51, 204, 0);stroke:black;stroke-width:1; }  
  
#Uşak{fill:rgb(204, 153, 51);stroke:black;stroke-width:1; }  
  
#Aksaray{fill:rgb(255, 204, 0);stroke:black;stroke-width:1; }  
  
#Diyarbakır{fill:rgb(255, 204, 0);stroke:black;stroke-width:1; }  
  
#Batman{fill:rgb(153, 102, 51);stroke:black;stroke-width:1; }  
  
#Denizli{fill:rgb(255, 255, 204);stroke:black;stroke-width:1; }  
  
#Kahramanmaraş{fill:rgb(255, 255, 204);stroke:black;stroke-width:1; }  
  
#Isparta{fill:rgb(255, 102, 102);stroke:black;stroke-width:1; }
```

```
#Niğde{fill:rgb(153, 255, 255);stroke:black;stroke-width:1; }  
  
#Siirt{fill:rgb(255, 204, 153);stroke:black;stroke-width:1; }  
  
#Adana{fill:rgb(192, 192, 192);stroke:black;stroke-width:1; }  
  
#Aydın{fill:rgb(255, 153, 255);stroke:black;stroke-width:1; }  
  
#Adıyaman{fill:rgb(255, 255, 204);stroke:black;stroke-width:1; }  
  
#Hakkari{fill:rgb(255, 153, 255);stroke:black;stroke-width:1; }  
  
#Şanlıurfa{fill:rgb(255, 255, 153);stroke:black;stroke-width:1; }  
  
#Şırnak{fill:rgb(51, 204, 0);stroke:black;stroke-width:1; }  
  
#Burdur{fill:rgb(255, 255, 153);stroke:black;stroke-width:1; }  
  
#Muğla{fill:rgb(153, 255, 153);stroke:black;stroke-width:1; }  
  
#Mardin{fill:rgb(255, 153, 0);stroke:black;stroke-width:1; }  
  
#Osmaniye{fill:pink;stroke:black;stroke-width:1; }  
  
#Karaman{fill:rgb(255, 204, 0);stroke:black;stroke-width:1; }  
  
#Gaziantep{fill:rgb(153, 153, 153);stroke:black;stroke-width:1; }  
  
#Antalya{fill:rgb(51, 204, 0);stroke:black;stroke-width:1; }  
  
#Mersin{fill:rgb(153, 102, 51);stroke:black;stroke-width:1; }  
  
#Kilis{fill:green;stroke:black;stroke-width:1; }  
  
#Hatay{fill:rgb(255, 153, 255);stroke:black;stroke-width:1; }
```


7.3 CSS Properties

There are some properties of CSS2 which CSS technology has. Each of these properties provide visual change in SVG.

Font Properties: font, font-family, font-size, font, font-size-adjust, font-stretch, font-style, font-variant, font-weight

Text Properties: direction, letter-spacing, text-decoration, unicode-bidi, word-spacing, alignment-baseline, baseline-shift, dominant-baseline, glyph-orientation-horizontal, glyph-orientation-vertical kerning, text-anchor, writing-mode.

Media Properties: color, cursor, display, overflow, visibility.

Common basic properties defined at SVG vector elements.

Color Properties: color-interpolation, color-interpolation-filters, color-profile, color-rendering.

Boundary Properties: stroke, stroke-dashoffset, stroke-dasharray, stroke-linejoin, stroke-miterlimit, stroke-width, stroke-opacity.

Image and shape Properties: image-rendering, shape-rendering, shape-dasharray, text-rendering.

Pointer Properties: marker, marker-start, marker-end, marker-mid.

Fill Properties: fill, fill-opacity, fill-rule.

Filter Properties: enable-background, filter, flood-color, flood-opacity, lighting-color.

Gradient Properties: stop-color, stop-opacity.

Interactivity Properties: pointer-events.

SVG vector element is gained style property by style which wanted to give alternatively using as attribute value. As an example, the following example can be examined.

```
<circle stroke="#000000" fill="#00ff00" />
```

It is known that same style properties are written like at the following example with inline-coding and CSS.

```
<circle style="stroke: #000000; fill:#00ff00;" />
```

CSS selectors which is defined in parallel work are applied on Point, Line and Polygon geometry element like at the below.

```
<polyline class="polyline" points="{ $clist }"></ polyline>
```

CSS definition of the “polyline” class is that

```
.polyline{ fill:blue;stroke:gray;stroke-width:2;}
```

This expression is defined inside of the `<style type="text/css"></style>` definition block. For other two elements, classes are obtained to each geometry element with same definition technics.

```
Class: .path{fill:none;stroke:black;stroke-width:1}
```

```
Geometry element: <polyline class="path" points="{ $linearRingCoordl2 }"/>
```

And other pair is like at the below.

```
Class: .point{fill:pink;stroke:black;stroke-width=1;}
```

```
Geometry element: <circle class="point" cx="{ $xCoord }" cy="{ $yCoord }" r="6.417" />
```

8 Client Side Scripting With Javascript and EcmaScript

Before JavaScript is mentioned as a programming language, let us discuss some properties of script languages. Script languages is a technology is that allows preparation of interactive web pages with allowing change of the HTML source codes for web developers. JavaScript programming language uses ECMAScript-262 standard. It is based on C language and it is an Object-oriented programming language. It was developed by the Netscape firm in 1995. Codes which are written in JavaScript language only work with Web Browser. The browser interprets the codes written in JavaScript and the browser will display on the screen.

The difference from other programming languages in the JavaScript language is that is not a run file like a “com” or an “exe”. It services just on the internet. Commands which are written with JavaScript can give different results in different browsers. These commands serve as a part of the HTML pages. In addition, these commands allow the management of the embedded SVG into HTML. SVG elements which are produce as static are handled as A DOM component. And then JavaScript is used on this element for adding interactivity to these elements. In the same way, SVG commands which are embedded in files which are prepared with XSLT are styled with CSS. And then these commands have interactive properties using EcmaScript. By using EcmaScript, it is possible to programme processes on these SVG files.

The scopes of the thesis are the encodings with EcmaScript codes and are mentioned here.

8.1 For locating North on a map

With the defining onclick="pan (0,- 50)" event on the map that is drawn with SVG, North is located on a map. These “pan” codes which are written with ecmaScript are below.

```

var transMatrix = [1,0,0,1,0,0];

function pan(dx, dy) {

    transMatrix[4] += dx;

    transMatrix[5] += dy;

    newMatrix = "matrix(" + transMatrix.join(' ') + ")";

    mapMatrix.setAttributeNS(null, "transform", newMatrix);

}

```

8.2 For locating South on a map

This is the same method for routing. On the maps which is drawn with SVG with `onclick="pan(0, 50)"` an event it is possible to locate South on a map.

8.3 For Zooming

The block of code that is required to be able to zoom the map is as follows. With this code block, by using `onclick="zoom('1.25')"` the event map is increased in size by 25% and with using `onclick="zoom('0.75')"` the event map is smaller by 25%.

```

function zoom(scale) {
for (var i=0; i<transMatrix.length; i++){
transMatrix[i] *= scale;
}
transMatrix[4] = transMatrix[4] + (1-scale) * (width/2);
transMatrix[5] += (1-scale)*height/2;

newMatrix = "matrix(" + transMatrix.join(' ') + ")";
mapMatrix.setAttributeNS(null, "transform", newMatrix);
}

```

8.3 For Entire Layer

It is possible to return map's first download situation. To manage this operation the following codes are included to projects.

```

function EntireLayer(){
    transMatrix[0] = 1;
    transMatrix[1] = 0;
    transMatrix[2] = 0;
    transMatrix[3] = 1;
    transMatrix[4] = 0;
    transMatrix[5] = 0;
    newMatrix = "matrix(" + transMatrix.join(' ') + ")";
    mapMatrix.setAttributeNS(null, "transform", newMatrix);
}

```

In addition, a lot of the functionality is possible by using jQuery which is a javascript library. For example, when one clicks anywhere on the map, it is possible to visualize the name of the clicked on place. The code is;

```

$(".path").click( function () {$("#Names").html($(this).attr("id"));});
$(".point").click(function () {$("#Names").html($(this).attr("id"));});
$(".polyline").click(function () { $("#Names").html($(this).attr("id"));});

```

By using `$(this).attr("id")` code, it is possible to get ID attribute of the clicked element. Thus, the ID value to “Names” ID named object are sent.

Another example is a jQuery method which is obtained CSS information of clicked object can be given at the below.

```

$(".polyline").click(function () {
    $("#layerproperties").css('visibility', 'visible');
    var renkkodu = $(".polyline").css('fill');
    $("#fillcolor").html(renkkodu);
    $("#fillcolor2").css('background-color', renkkodu);
    var cizgikodu = $(".polyline").css('stroke');

```

```
$("#linecolor").html(cizgikodu);  
$("#linecolor2").css('background-color', cizgikodu);  
var cizgigen = $(".polyline").css('stroke-width');  
$("#linewidth").html(cizgigen);  
var linedash = $(".polyline").attr('stroke-dasharray');  
$("#linedash").html(linedash);  
var opacity = $(".polyline").css('opacity');  
$("#opacity").html(opacity); });
```

9 Web Feature Services (WFS)

Today's society is progressing rapidly towards becoming a mobile society. For this reason, it is inevitable that any used information be used to support staff. Obtaining location information is a very long and laborious a process. For this reason, part of it is still on paper, while a part of it has been digitalized. This information has been collected for the past twenty years. After this type of geographical information is digitalized, it is stored in different forms on computer according to the purpose of the service. The storing process is realized in database systems and information technology platforms. The Open Geospatial Consortium (OGC) organization was established on the 25th of September 1994 for spatial information standardization. It is a community that determines all geographic standards. This community has published standards which are most commonly used in Geographic Information Systems, “Web Map Service (WMS)” and “Web Feature Service (WFS)”.

9.1 OGC Web Services

First of all, brief information about the web services will be given. A web service is an application service. A web service is created by one or more functions coming together. Web services are independent of the operating system and programming languages. These service functions have characteristic properties such as each of them has to fulfil a specific task. As an example, a given a web service function named “generate map“. This function, as the name suggests, generates maps. The client accesses remotely to the functions which are in web services. In web services, format of input and output parameters is XML (Extensible Markup Language) format. There are three “open web services” in Geographical Information Systems. These open web services also are named as “OGC Web services”. These services work with geographic information. So, these services are an identification of the standard interface, protocols, encodings and schemas. (Fallman, 2004) OGC web services service on the internet platform with using the http protocol.

OGC web services are Web Map Service (WMS), Web Feature Service (WFS) and Web Coverage Service (WCS). WMS (Web Map Service) is the most widely used service.

A feature which distinguishes the map services is the returned format of the maps. There are two types of formats, including vector and raster. In raster map format, maps can be any image format, on the other hand in vector format, maps can be expressed in vector objects.

If we want to describe in more detail, raster data format represents a coordinate system. Each square covers a significant geographical area. These areas are called cells. Raster cells are the smallest map unit of GIS systems. The size of this unit affects the resolution directly and resolution is increases when get the smaller. It is achieved the vector format data from the bird's eye view of the real world and modelling. The actual situation representation is made by vector objects. These objects such as vector polygon, line, point, refer to all objects. In Geographical Information Systems, it is preferred that getting vector format output.

In the following example, the difference between a raster and vector map format can displays good.

Below, there is an example that the same in both raster map and vector map.

R= Road, L= Land, H= Home, B=Beach C= Conservation area, S = Sea

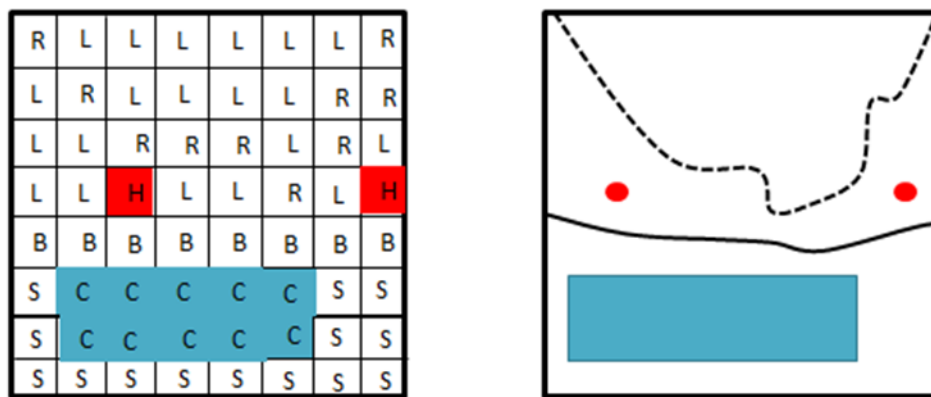


Figure 9.1.1 , The difference between a raster and vector map format

9.2 Web Map Services (WMS)

In the web map services (WMS), result maps are raster picture format of map object. This map object is not a data on its own property. There are many picture formats. These are PNG, GIF, JPEG, SVG and WebCGM. WMS is descriptive service type. This service type just can make presentation on graphical and non-graphical formats. It is reached to geographic information and the information about the server with various operations.

9.2.1 Web Map Services (WMS) Operations

There are three operations which is used on WMS (Web Map Service) service specifications. The first of these two operations are necessarily used operations. In short, if we are to talk about;

GetCapabilities: It is a necessary operation for reaching metadata which is related with service provider that store geographical information. The query statement is as follows.

<http://hostname/cgi-bin/mapserv?map=mapfile.map&VERSION=1.1.1&REQUEST=GetCapabilities>

After this query is worked on, a response is sent in XML (Extensible Stylesheet Language) format. This response contains the datum, graphic data structure, size etc.

GetMap: It is process step which is the most basic step in the processing of the query which is returned a map from Web Map Service. The following query is provided with this transaction.

“<http://hostname/cgi-bin/mapserv?map=mapfile.map&VERSION=1.1.1&REQUEST=GetMap&LAYERS=cities&BBOX=x,x,x,x>”

We saw the picture below, in Figure 9.2.1.1, it is a jpeg formatted map. This map is obtained from “GetMap” query.



Figure 9.2.1.1 A jpeg formatted map (Accessed through Internet)

GetFeatureInfo: It is a query that retrieves map-related properties from Web Map Services. This query don't necessary for visualizing of the map and it is depends on the user's request. Result of the query is contains information that is being queried. In the query, coordinates that is desired information are send to query statement and then it is retrieved attribute informations.

9.3 Web Feature Services (WFS)

Result vector maps in Web Feature Services (WFS) returns spatial information in GML format as attribute information and the resulting data type is a service that allows you to access the contents of the data. For the scope of this thesis, Web Feature Services (WFS) is preferred. It is used in OGC (Open Geospatial Information) specifications for transportation and manipulation operations of geographic data that crafted with the Geographic Markup Language (GML). WFS services is a necessary technology to get geographic information in GML. WFS services is using for GIS support which it obtains a GML data for the GIS system. A WFS service provider takes, reads, analyses requests and then sends the result back as the GML data format. Files received from multiple sources can be combined into a geographic information System with GML data structure. If we consider background of these three web services that created by Open Geospatial Consortium (OGC), we can see two problem of the background. These services are developed for a solution of these two problems. The first of these problems is that interoperability problem; another problem is the addressing problem. Interoperability problem an emerging problem is that different organizations create your own data in different formats with

Geographic Information Systems. Addressing problem is that due to the existence of more than one server, each server due to the mixing of information.

9.3.1 Web Feature Services (WFS) Operations

A WFS service provider consists of querying and manipulation interfaces. The scope of the WFS services, there are three query operation which can be inquire about the WFS server. These are “GetCapabilities”, ”DescribeFeatureType” and “GetFeature”.

GetCapabilities: GetCapabilities method has informations about the term and operations of geographical data provided by the service provider. The resulting server response that the structure is XML contains appropriate and accessible feature types and the supported operations that geographic data can be executed. In addition there is an XML Schema which contains data type definitions that is returning properties in the query result. GetCapabilities request works with HTTPGET and HTTPPOST methods. It is possible to reach many properties of service provider with using this method. These properties are like this; “Server Name”, “Server Title” information which are related to “service” information. “Name”, “Title”, “Abstract”, “Keywords”, “Source”, “LatLongBoundingBox” information which are related to “Feature Type” information and GetCapabilities method also contains a lot more filtering capabilities and interrogation operations.

```
<?xml version="1.0" encoding="ISO-8859-1"?>  
<GetCapabilities version="1.0.0" service="WFS"/>
```

DescribeFeatureType: It is possible to reach more detailed information about property types which is handled with GetCapabilities method with using DescribeFeatureType method. Beside this, it is possible to reach other possible feature type’s details with defining feature type name. In the returned response from “DescribeFeatureType”, there are various properties about “FeatureType”. Usage of this properties and methods are mentioned below.

```
<?xml version="1.0" encoding="iso-8859-1"?>
```

```
<DescribeFeatureType outputFormat="XMLSCHEMA">
  <TypeName>cities</TypeName> ...
</DescribeFeatureType>
```

GetFeature: With sending a “Get Feature” request, it is possible to get “feature information” property information. The Response from “GetFeature” request is in a GML format. In the Response, all geographical data and non-geographical data are located in the “FeatureCollection” node.

```
<?xml version="1.0" encoding="iso-8859-1"?>
<wfs:GetFeature outputFormat="GML2" xmlns:gml="http://www.opengis.net/gml"
  xmlns:wfs="http://www.opengis.net/wfs" xmlns:ogc="http://www.opengis.net/ogc">
  <!--get all features of feature type cities and all their properties-->
  <wfs:Query typeName="cities">
    </wfs:Query>
  </wfs:GetFeature>
```

10 MAPXTREME Development Toolkit

MapXtreme is developed for software developers from Mapinfo GIS application which has wide user audience. It is a software development product that is ploved to the market to create effective web pages. With MapXtreme technology, a bridge is established between the geographic information systems and information systems. It not necessary that the data getting from web pages that developed with MapXtreme technology is constructed in tabular form, consisting of rows and columns. MapXtreme as a GIS software provides to present maps and positional data to the user. MapXtreme GIS-based software offers the positional information for all data processed as information to users on the map. MapXtreme has powerful geographical processing properties and has handy user-interface. MapXtreme is a server side development tool. So, it is named as “development kit”. MapXtreme development kit is preferred to develop web based GIS (Geography Information Systems) that produce special map on the .Net platform.

MapXtreme map pages development kit is developed on web based at the same time it is developed on desktop application. MapXtreme applications work on a managed server network named as “web server”. Defined “web server” must has a property that is having administrative options. In this thesis, Microsoft Server Manager version is used for administrative options. MapXtreme SDK has tool which is developed for desktop and web applications. Because MapXtreme SDK is .Net based and a fully compatible library it can provide flexible solutions for application development and easy installation. In addition, it can work compatible with databases and it can read and write geographical data over the database system. Reporting processes and querying processes on the map can make easily with this technology.

11 Configuration Settings

Configuration settings for presentation of geographical information in WFS service provider are required. Configurations settings are located in two configuration files. One of these files is “Web.config” file which contains all configuration settings related with projects inside another file is “WFSService.xml” configuration file which is constructed by project itself. This file is located in “WFSserver” file which is upper directory file. “WFSserver” is also constructed by project. This is located in C directory on local of the server computer. In the “WFSService.xml” file, there are WFS Service names. In the thesis, these processes are met codes are as follows.

```
static string directoryPath = @"C:\WfsServer\";
static string fileName = "WfsService.xml";
static string path = directoryPath + fileName;
        protected void XMLDataBind() {
                if (!Directory.Exists(directoryPath)) {
Directory.CreateDirectory(directoryPath); }
                if (!File.Exists(path))
                {FileStream fs = File.Create(path);
                fs.Close();
                FileStream Servicexml = new FileStream(path, FileMode.Create);
                XmlTextWriter xml = new XmlTextWriter(Servicexml, Encoding.UTF8);
                xml.WriteStartDocument();
                xml.WriteStartElement("WfsAddress");
                xml.WriteEndElement();
                xml.Close();
                return; }
                using (DataSet ds = new DataSet())
                {ds.ReadXml(path);
                if (ds.Tables.Count > 0)
                {DropDownList1.DataSource = ds;
                DropDownList1.DataTextField = "WfsName";
                DropDownList1.DataValueField = "URLName";
                DropDownList1.DataBind();}
                } }
```

In the “WfsService.xml” file, there are WFS service name which are appropriate for the project and URLs. In the “Web .config” standard settings file, there are Asp.Net configurations which are related with the project. There are 3 main sections in configuration of the project. In the first section, there are WFS server name and WFS server URL in the configuration file.

```
<configuration> <appSettings>
  <add key="configFile" value="C:\wfs\WFSSample.xml" />
  ...
</appSettings></configuration>
```

The second section is done with the following settings. In this section user or developer should update these settings according to IIS version of working computer.

```
<system.web>
  <httpHandlers>
<add verb="GET,POST" path="*.ashx"
type="MapInfo.Wfs.WfsHttpHandler, MapInfo.Wfs.Server, Version=7.1.0. 228,
Culture=neutral, PublicKeyToken=93e298a0f6b95eb1"/>
  </httpHandlers>
  <httpModules>
<add type="MapInfo.Engine.WebSessionActivator,
MapInfo.CoreEngine, Version=7.1.0. 228, Culture=neutral,
PublicKeyToken=93e298a0f6b95eb1" name="WebSessionActivator" />
  </httpModules>
</system.web>
```

(For additional Assembly information, see
“C:\Windows\Microsoft.NET\assembly\GAC_32” or
“C:\Windows\Microsoft.NET\assembly\GAC_64”)

```

<system.webServer>
<validation validateIntegratedModeConfiguration="false"/>
<directoryBrowse enabled="true" />
<modules>
<add name="WebSessionActivator" type="MapInfo.Engine.WebSessionActivator,
MapInfo.CoreEngine, Version=7.1.0. 228, Culture=neutral,
PublicKeyToken=93e298a0f6b95eb1" />
</modules>
<handlers>
<add name="WFSHandler" verb="GET,POST" path="*.ashx"
type="MapInfo.Wfs.WfsHttpHandler, MapInfo.Wfs.Server, Version=7.1.0. 228,
Culture=neutral, PublicKeyToken=93e298a0f6b95eb1"/>
</handlers>
</system.webServer>

```

In section two, WFS service providers are identified in the “WfsService.xml” file. The structure of “WfsService.xml” as below.

```

<?xml version="1.0" encoding="utf-8"?>
<WfsAddress xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <Address>
    <WfsName>Pitney Bowes</WfsName>
    <URLName>http://www.mapinfo.com/miwfs</URLName>
  </Address>
  <Address>
    <WfsName>MetroGIS Water Resources</WfsName>
    <URLName>http://gis2.metc.state.mn.us/ArcGIS/services/MetroGIS/Water_Resources/MapServer/W
FSServer</URLName>
  </Address>
  ...
</WfsAddress>

```

In the last section, after constructing “Web.config” and “WFSService.xml” files, WFS Service Provider Internet Information Service (IIS) should be constructed. It is important that MapXtreme technology can works under the Microsoft Wow

(Windows on- Windows 64-bit). Default download directory of MapXtreme is C:\Program Files (x86) \MapInfo\MapXtreme\7.x.x". For this reason, there are two paths for "workspace" value which is defined on the "web.config".

```
<add key="MapInfo.engine.session.Workspace" value="C:\Program  
Files\MapInfo\MapXtreme\7.0.0\Samples\Data\World.mws; C:\Program Files  
(x86)\MapInfo\MapXtreme\7.0.0\Samples\Data\World.mws"/>
```

12 Related Works

It is possible to give an example which is similar with this research. For example, in 2001, in the University of New Caledonia, the work done by Gilles TALADOIRE. The name of the work is “Geospatial Data Integration and Visualisation Using Open Standard”. In the study, it has been referred to be displayed with the GML open standard. It is mentioned that GML files, XML schemas and XSLT technologies and works are realized by these technologies. (TALADOIRE, 2001)

In 2003, “Visualization of GML data using XSLT” named research was conducted in Germany. The focus of this research was the interoperability of the applications and on visualizing GML data with XSLT technology and as related to this topic, visualization of GML data with the provided XSLT. During the process of displaying, CSS technology and client-side coding method also took place. (Tennakoon, 2003)

In 2003, in Italy, it is mentioned that advantages of using GML data format in “SVG Explorer of GML Data” named work. It is also referred that it is possible that display through XSLT with SVG. It is described to acquisition process of GML data with WFS (Web Feature Service) services. **(Bonati, 2003)**

An application which is developed in 2003, named as “Publishing GML data as interactive SVG maps”. This application display GML data as SVG format. Work is realized and developed by software specialist Alison Meynert. Technologies which are located in the development of technologies for the application are XSLTC application which belong to Apache and Java. Translation of GML data to SVG is realized with java codes. In the XSLT stylesheet files, was used, with XSLT extension functions (Saxonica Extended Functions, 2014). Within the application, GML data converted to SVG format dynamically, on the fly. (Meynert, 2003)

Another research work, in Finland, a work titled with “Schema Translations by XSLT for GML-Encoded Geospatial Data in Heterogenous Web-Service Environment”. The work was done in 2004. As a solution of interoperability, it is pointed out that visualization of GML data with XSLT schema in heterogeneous web services. (Lehto and Sarjakoski, 2004).

Another survey conducted in China in 2004, is named as “Modeling GML and SVG Data for Web GIS”. The survey, was handled by Liang ZOU. In the parallel to this survey, it tried visualization of GML data. (Liang Zou, 2004)

In 2006, a work which is researched as master thesis, “GML processing with XSLT and spatial extensions”; the aim was to write a GeoXSLT that attempted to translate data files which contain geographical information encoded with GML. (Fredrik and Klausen, 2003).

13 Application

This thesis is developed with using Mapxtreme development kit for downloading WFS services. The programming language of the thesis is C# programming language on the .Net platform. Mapxtreme is powerful development kit for developing web based geography information system applications. These applications can be applications that working with variety of vector data formats like GML data format. GML has a lack of data imaging ability. Because of this, to gain this ability it can be useful to use XSLT processor technology. In this thesis's application, Microsoft XML technology was used, as was Saxon technology. And Microsoft XML technology was found to be preferred. Saxon processor is a technology that works as XSLT processor. Project has three sections. One of the sections is that WFS Server sections. In this section, WFS server connection is made by the user. If it is wanted, other new connections can add the system. After connecting the WFS server, GML data is obtained by the connection. This GML data is saved on the C directory. This GML data needs to manipulating process. After manipulating process it become available for interpreting. With using prepared general XSLT, it is possible to display text based GML data to two dimensional SVG format. In section two, it is processed with .WMS files which are obtain from "Workspace Manager" application.



Figure 12.1 The Application Mainpage's view

This application is related to Mapinfo application. "Workspace Manager" translates. TAB mapinfo files to .WMS file. In last section, there is GML viewer. Users can display GML files with using this section. GML files which are on the local machine

are visualized by the section. All explained sections are like at the below. In the last tab, there is information about the project. We will examine those sections respectively.

13.1 WFS Client

GML visualization application with using WFS service link is named as “WFS Client”. When page is first loading, if there is WFS service addresses saved before, these addresses are listed in the “dropdown list” object.

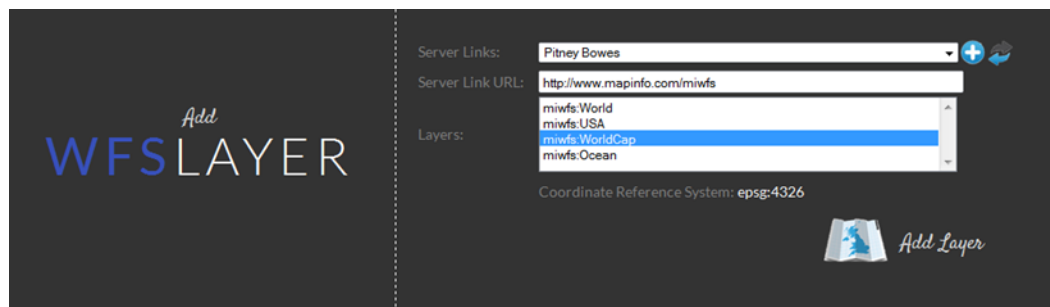



Figure 12.1.1 WFS Client Connection

If user want to create a new WFS server URL address in to dropdownlist, he can click the “add”  button. User can obtain to open a new popup window with clicking to the button. Then user can reach in the following screenshot.

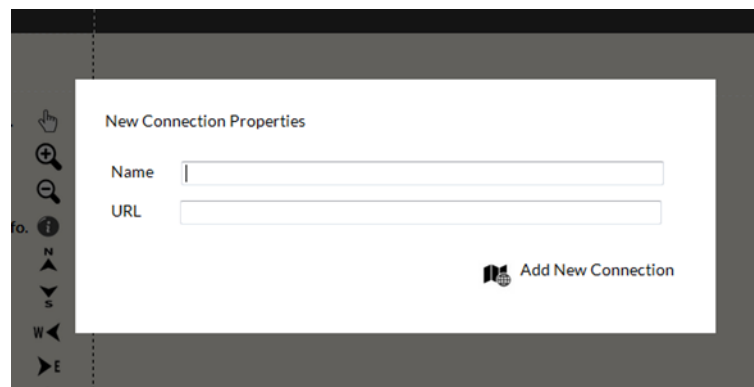



Figure 12.1.2 Adding a New Connection

In Figure 12.1.2, “New Connection Properties” named popup window is seen, there are two input boxes. One of these boxes is “Server Name” which is one of the

property of the new connection properties. Another is “Server URL” which is another new connection properties. “Server URL” property is the full path of the URI which user wants to connect. After entering these two value, click “Add New Connection” button. By this operation, it is constructed a new WFS server link and it is added WFS Server config file. For example, we connect <http://www.mapinfo.com/miwfs> address. It is seen that layers at the below listed. When click the one of them, it is seen that “epsg:4326” text on the right side. It is Coordinate Reference System of the layer. For example at the figure it is clicked that “miwfs: WorldCap” layer. When click the “Add Layer” button, if document is not buffered before, GML file is saved in to C directory on user’s computer. Buffered file is visualized as SVG format by using XSLT together. Screenshot of this process is like in the Figure 12.4. At the below, World Capitals are visualized with point object. At the same time World Capitals Names are added beside of the objects. If it is wanted to visualize just geometry set, it is possible to eliminate World Capitals Names with clicking  button .

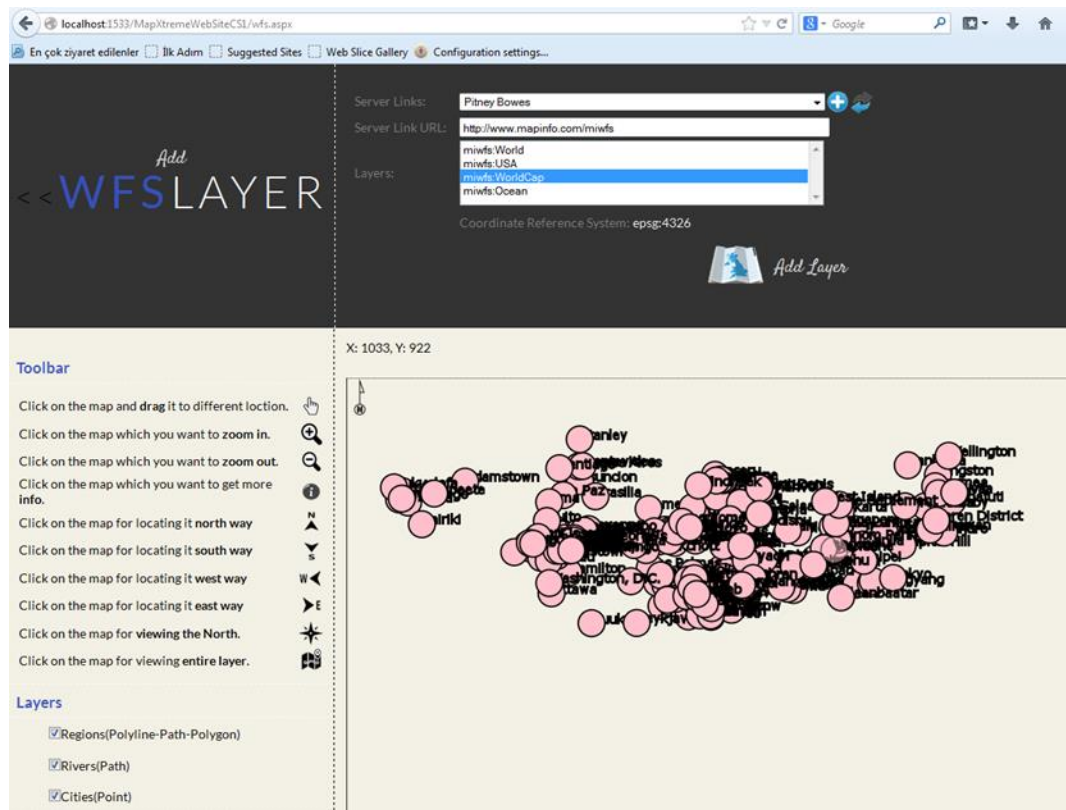


Figure 12.1.3 World Capitals

As another example, “miwfs:Ocean” layer which is obtained by clicking <http://www.mapinfo.com/miwfs> WFS address. Select “miwfs:Ocean” layer and then click on “Add Layer” button. Then it is reached layer that at the below Figure, Figure12.1.4.

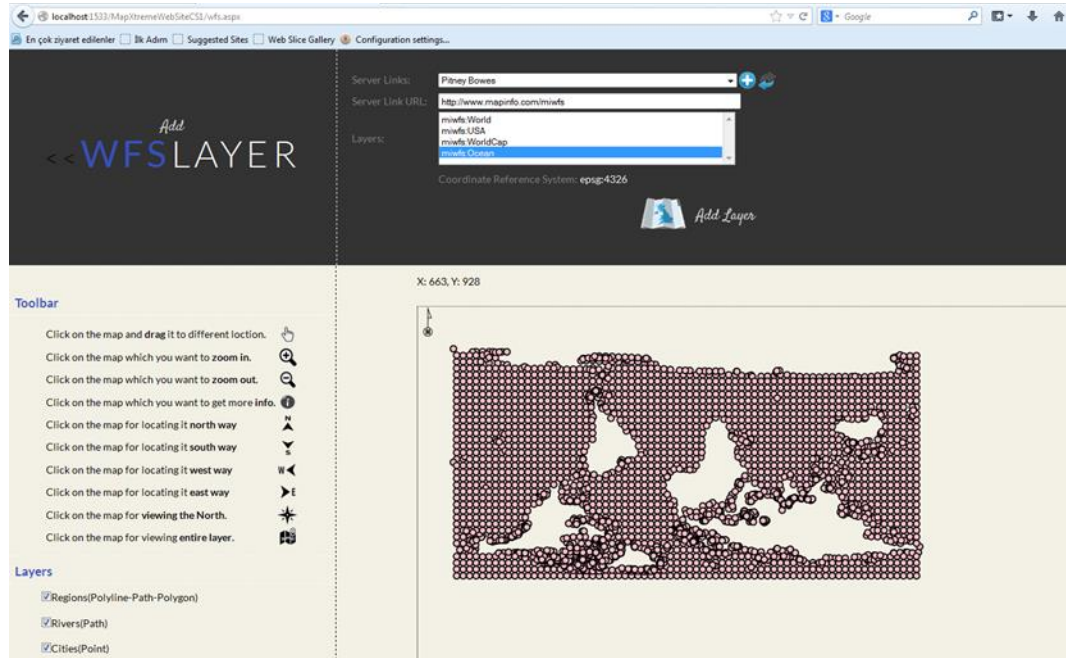


Figure 12.1.4 Oceans returned from WFS

As seen at the below Figure, there are three main section on the left side. In the first section there is a ToolBar is located. With this ToolBar, it is possible to zoom in, Zoom out, locating North side, locating South side, locating East and West. At the same time it is possible to show/hide North Notwithstanding and it is possible to make Entire Layer on the map. In addition with information button, user can visualize the related information of the objects. In section two, there are show hide buttons which is worked on geometry elements (polygon, point, path) on maps that produced programmatically. These tools are prepared for layers. It is found that showcase of prepared for three geometry elements in SVG like at the below, in Figure 12.1. 5

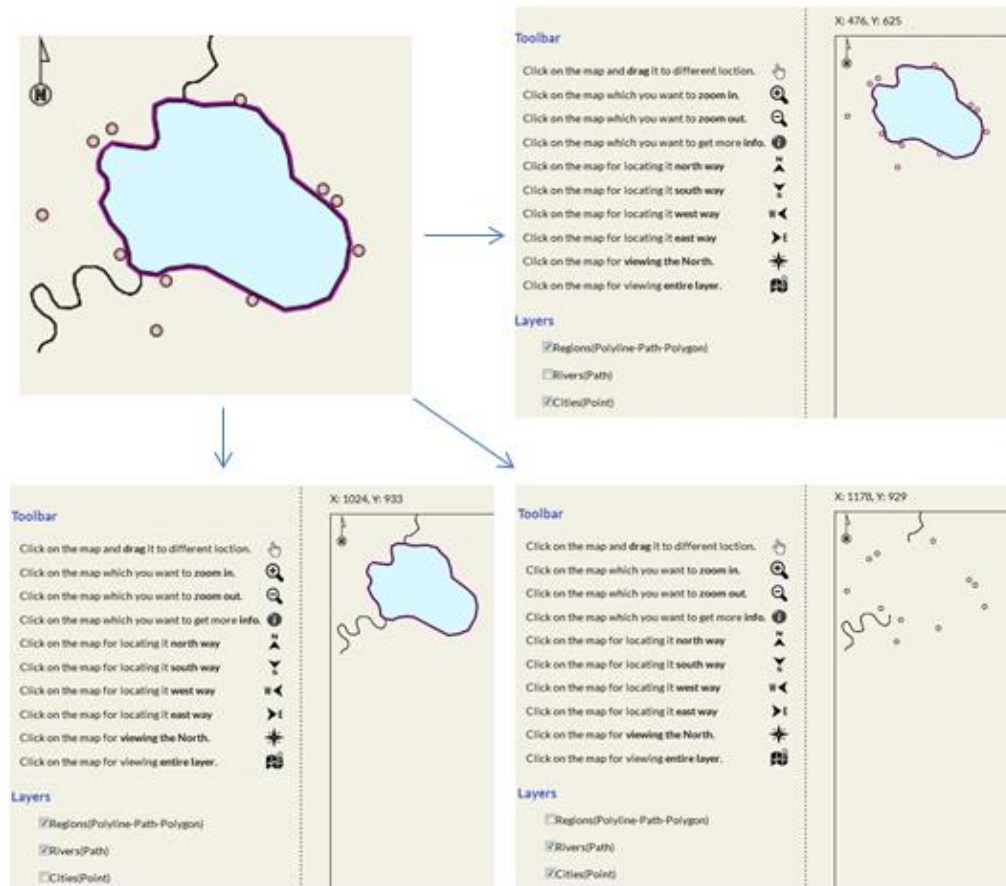



Figure 12.1.5 Layer Show/hide Property

Apart from these, in the third section there is named “Layer Properties” section. In this section, there is a panel which reflect the properties belonging to the geometry element. If it is wanted to set these properties by clicking th settings button it is managed to settings operation. This mentioned panel is show in Figure 12.7 at the below. When user click  button on the “POLYGON” section, he/she reached that following screenshot like at the below. In this step, neccessary settings are made by user and then clicking “Save” button. Settings are reflected to map.

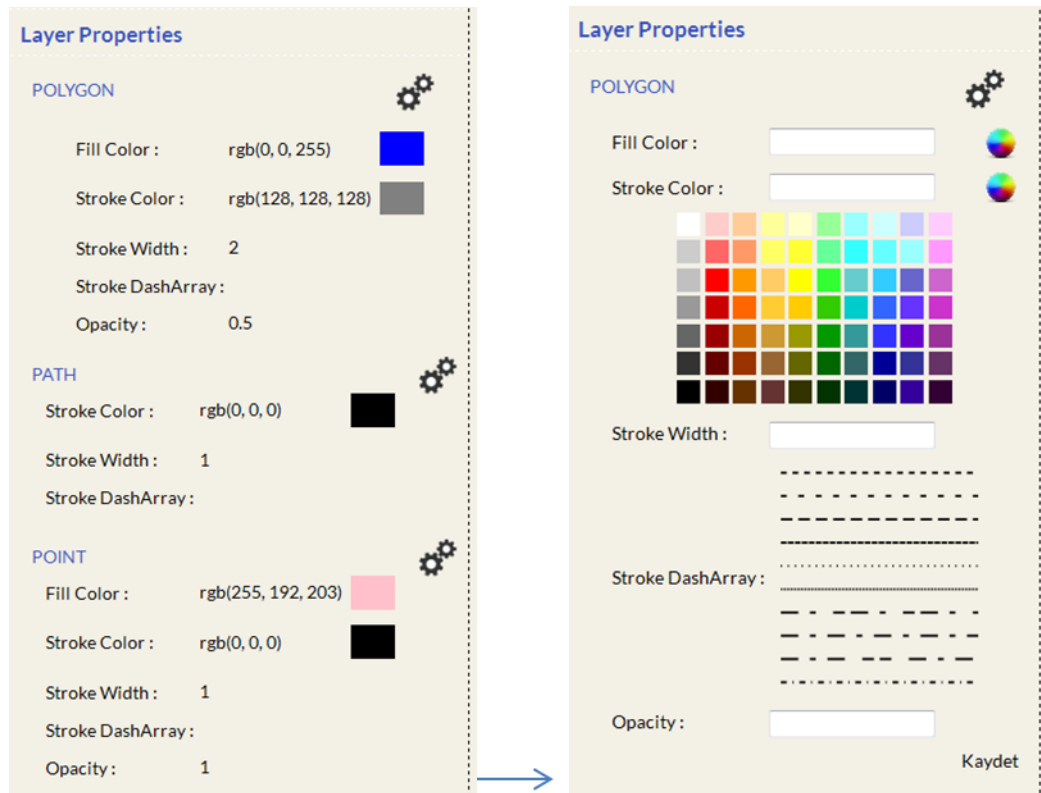


Figure 12.1.6 CSS Customization Tool

For example settings are the following;

FillColor: rgb(255, 102, 0)

StrokeColor: rgb(153, 51, 153) ,

Stroke Width: 3,

Stroke DashArray: (10,5)

And Opacity:1,5

These settings are reflected in the map and screenshot of the map are like below, in Figure 12.1.7.

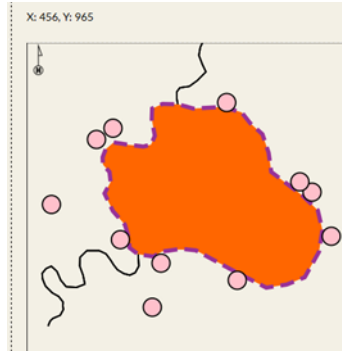


Figure 12.1.7 Customized Shape with the Tool

13.2 Mapinfo Tab Files Section

“Map Info Tab Files” section is located on second tab at the main page of the project. If it is clicked to this tab, the screen like below is reached. As it is seen on the screenshot “Turkey” map is retrieved as ready. There are many applications which is made on this map. At the same other page, there are some tools for the left side of the page. One of this tools is at the top of the map. In there, there is a tool which is list all city names of the map. By selecting one of the these cities it is possible to visualize a map of a selected city. The second tool is the “Layers” tool. In “Layer” tool, it is possible to Show/Hide layers. The third tool is named as “Labels”. This tool works on layers and it shows and hides labels which wanted to display. The fourth tool is “Theme Settings”. Themes are being created according to the values specified here. There are four different themes. These are “Regional Theme”, “Pie Theme”, “Bar Theme”, “Graduated Theme”. In the theme settings section, there is "Legend" option for created themes. In the last section, there is "Updates" part of the project Update part updates operations can be made on the map. Here, the layer is selected that is desired to update and updating of the map is provided by entering the old and new values.

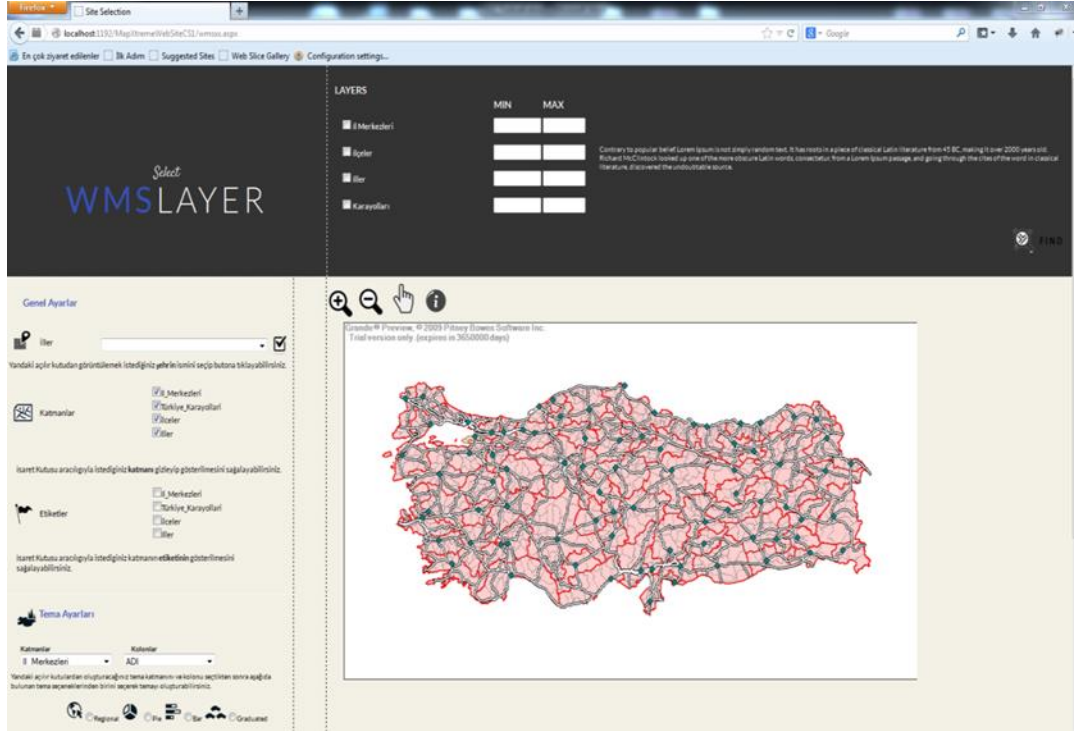


Figure 12.2.1 Turkey Tab File Visualization

For example, when İzmir city is selected and then it is clicked the button, like at the below it is reached the screenshot.

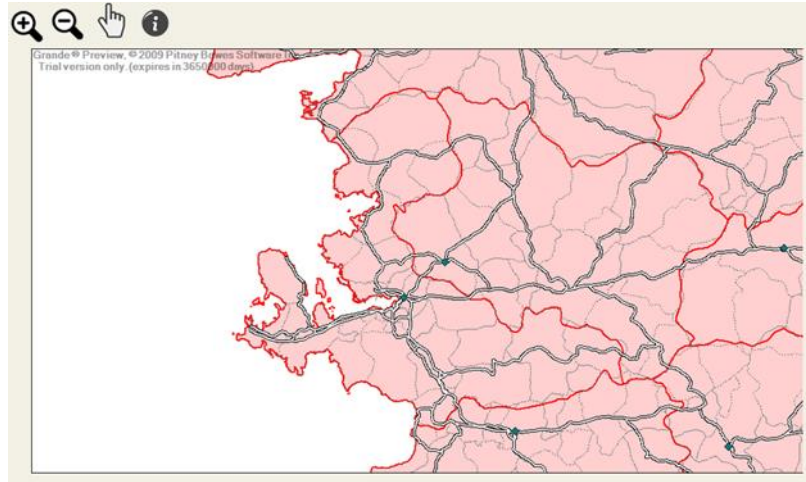


Figure 12.2.2 Selection of İZMİR on the Map

There is a list of geometry objects which is on the map in the “Layers” section. The list is listed as checkboxlist. It is possible to see showing and hiding when clicking on one of the these checks. When checking one of the checks, clicked geometric representation influences all the geometries on the map. For example, when unchecked “İlmerkezleri”,”Türkiye Karayolları” and “İlçeler”, there is just “İlsınırları” are showed. It will be like the following screenshot.

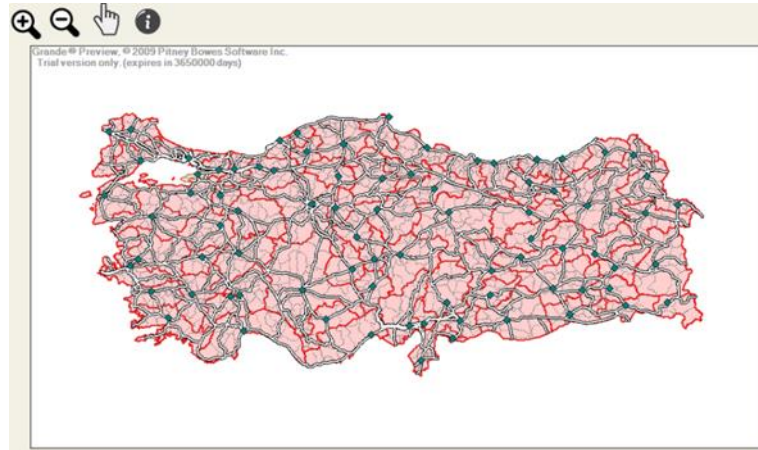


Figure 12.2.3 Turkey Map with Layers



Figure 12.2.4 Turkey Map without Layers

General Settings portion of the third tool is the "Tags" tool. With this tool, it is provided to show and hide the information on the labels which belongs to layers shown on the map. For example, in the above example, when it is desired to add to city names at the Turkey City map, achieved image is as follows:

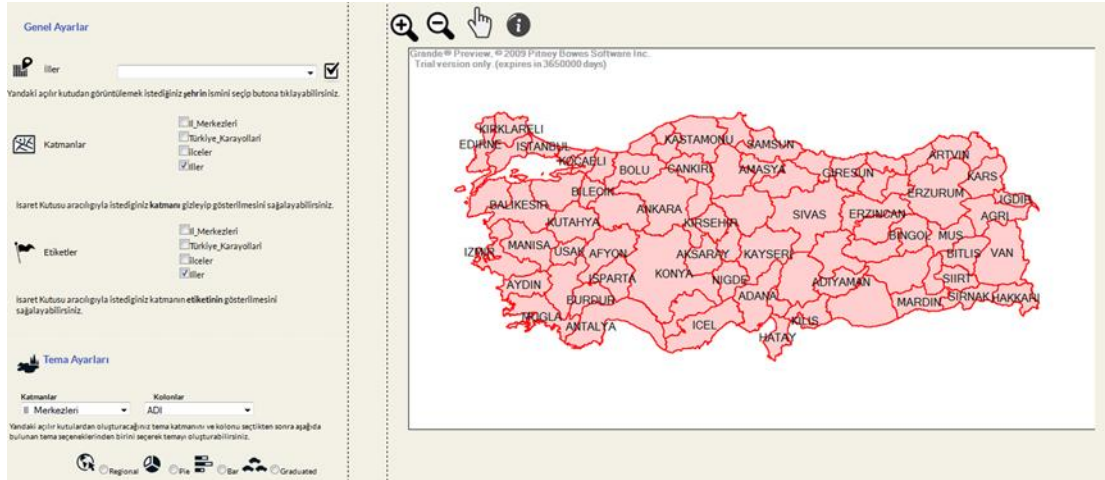


Figure 12.2.5 Labeling on the Map

“Theme Settings” section contains four different theme options. Theme options is shown below.



Figure 12.2.6 Theme options

For example, showcase of Provincial Centres according to “Nufus 1990” with “Regional Theme” like at the below.

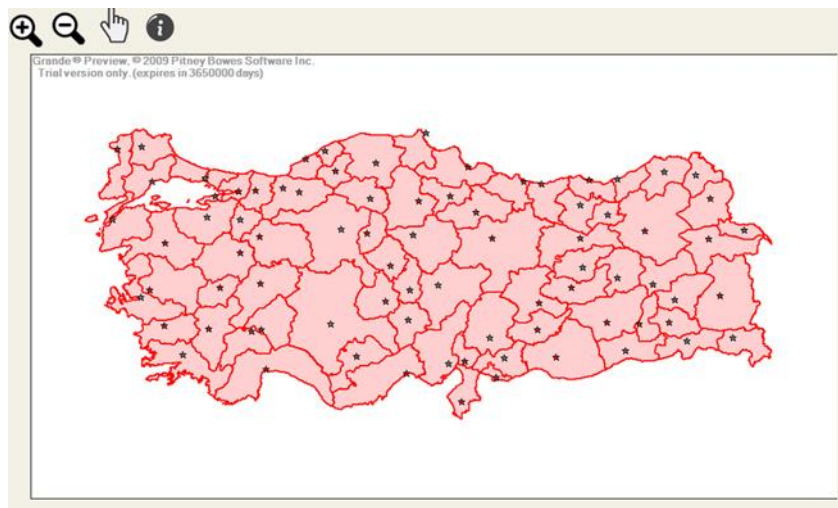


Figure 12.2.7 Regional Theme related to Turkey

Another example is that showcase of Provincial Centres according to” Nufus 1990” and “Nufus 1997” with “Pie Theme” ve “Bar Theme” like the following.

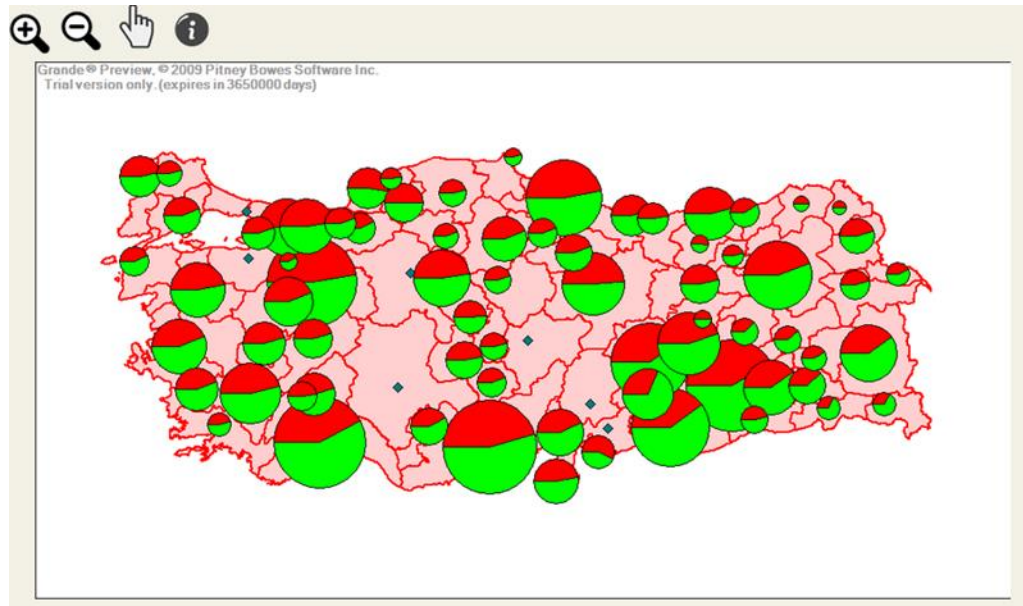


Figure 12.2.8 Pie Theme related to Turkey

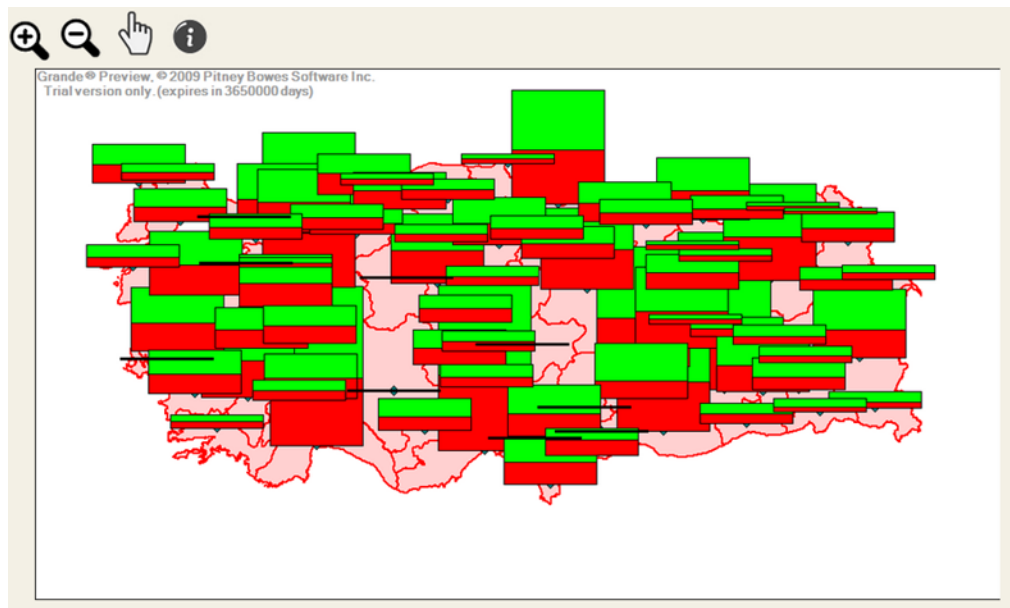


Figure 12.2.9 Bar Theme related to Turkey

The showcase of Provincial Centres according to” Nufus 1997” with “Graduated Theme” like below.

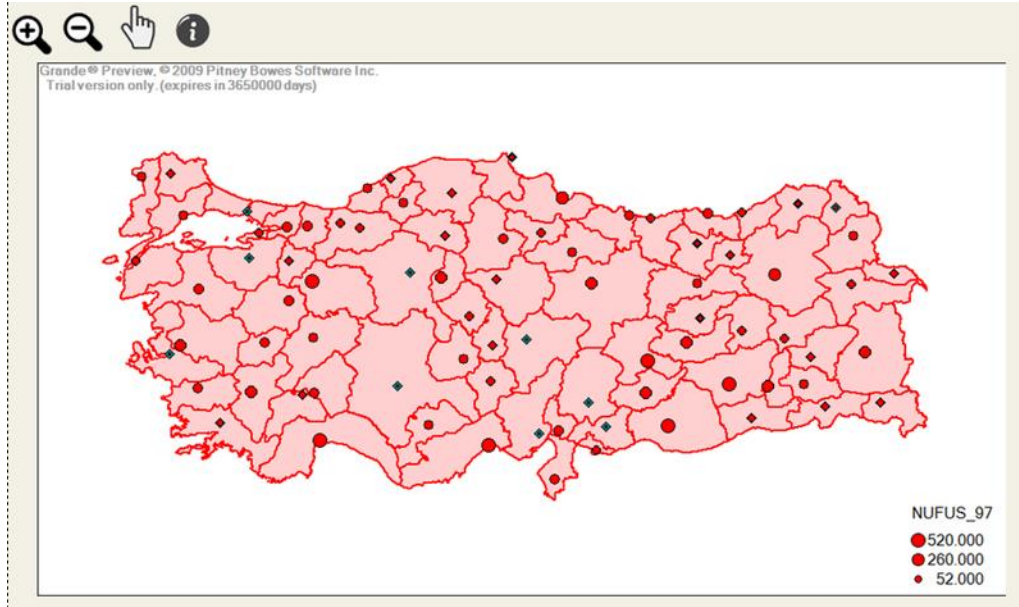


Figure 12.2.10 Graduated Theme related to Turkey

Finally, “Updates” section is a part of the section which will be data updates on the table data. In the section, there are four options. These options are “İl_Merkezleri”, “Türkiye_Karayollari”, “İlçeler” ve “İller”. One of the wanted to update options among these options is clicked. Then, select the coloumn which is want to update among these listed coloumn names. For example, as see at the below, in the screenshot “İller” layer was selected and was updated by entering new and old values related with the coloumn. So, updating process is realized.

Figure 12.2.11 Update Panel

GML View Section

In this section, it can be the visualized users' own GML files which are located on his/her computer. If users want, as in the screenshot below, a GML file with FileLoader can be selected.

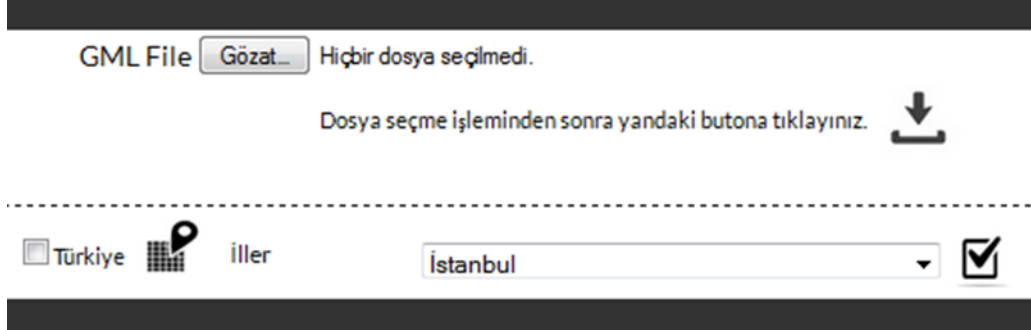



Figure 12.2.12 GML File Loed Panel

A file which is selected from the “Gözet” button is included with clicking the  button. For example, below, in Figure 12.2.2, the polygon image is displayed which belongs to Turkey.

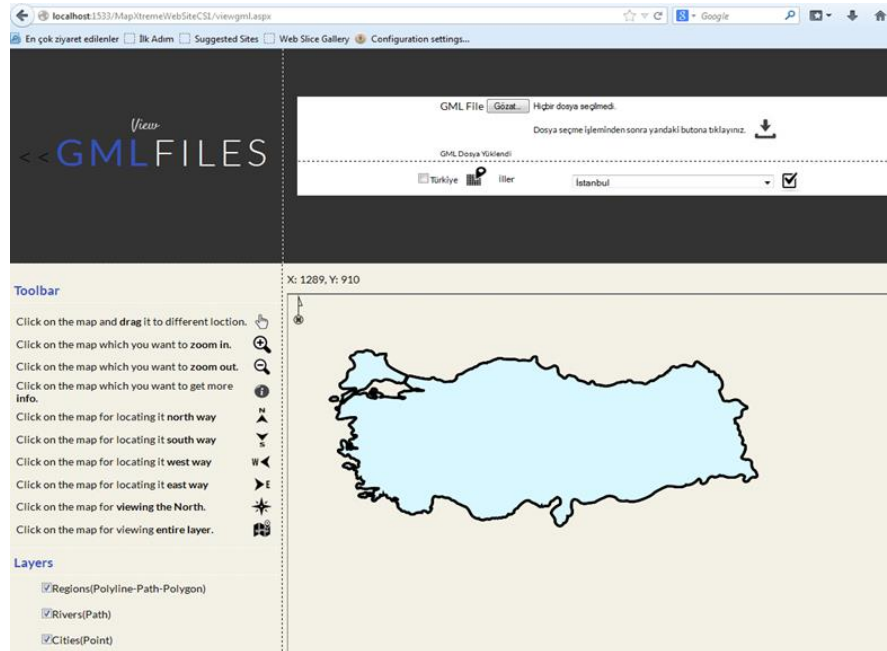


Figure 12.2.13 Blank Turkey GML visualization

With the help of the "Toolbar" which is on the left side, it is possible to make various kinds of visualizing operations (zooming, floating, viewing entire layer) on the displayed SVG elements. In "Layers" section it can be provided to Show and Hide of the geometry elements. At the below, in Figure 12.2.14, Turkey Provinces Map are demonstrated by reading GML file with XSLT on the browser.

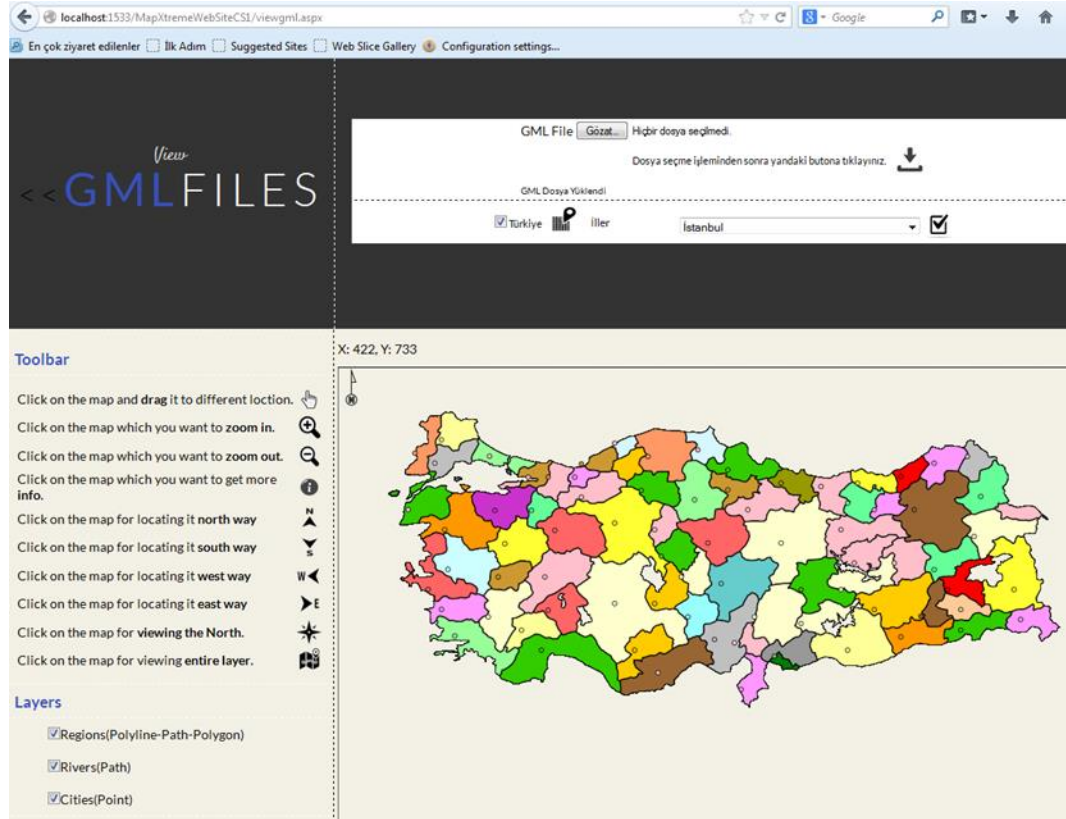


Figure 12.2.14 Turkey Provinces Map

In the map, we are able to click the Information Button and then we can obtain to display the names of the city. The showcase is in the Figure 12.2.14. In addition, we can get other Toolbar operations on the map. Map of the Turkey have classes when the map is loaded. It is possible to construct this map as one class and naturally same color. In addition it is possible to remove city boundaries, and show only the names of the province representing the points. For process of the step, it is necessary to uncheck "polygons" checkbox from "Layers" section. The showcase of the process is like at the below, in Figure 12.2.15.



Figure 12.2.14 Turkey Provinces Map with Label

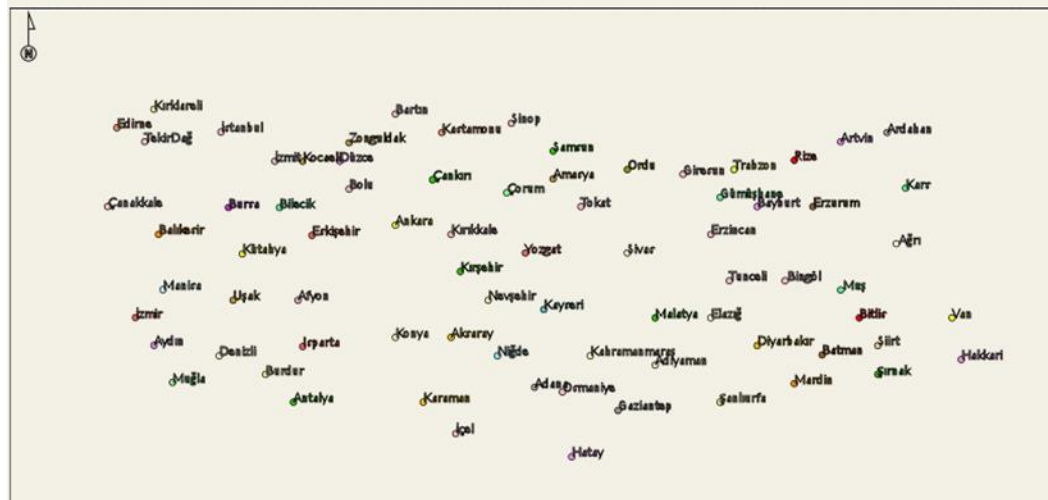


Figure 12.2.15 Provinces of Turkey

Instead of displaying the names of the entire province on the map, in addition there is an option for visualizing province information for clicked city. For example, below, there is a city which name is Ardahan. By clicking on a county which belongs to the province of Ardahan, the name of the County is shown in the upper right corner of the screen. With the operation, it is provided a graphical display of graphical information. Thus, all graphical information which is located in GML files and non-graphical information are reflected on the screen.

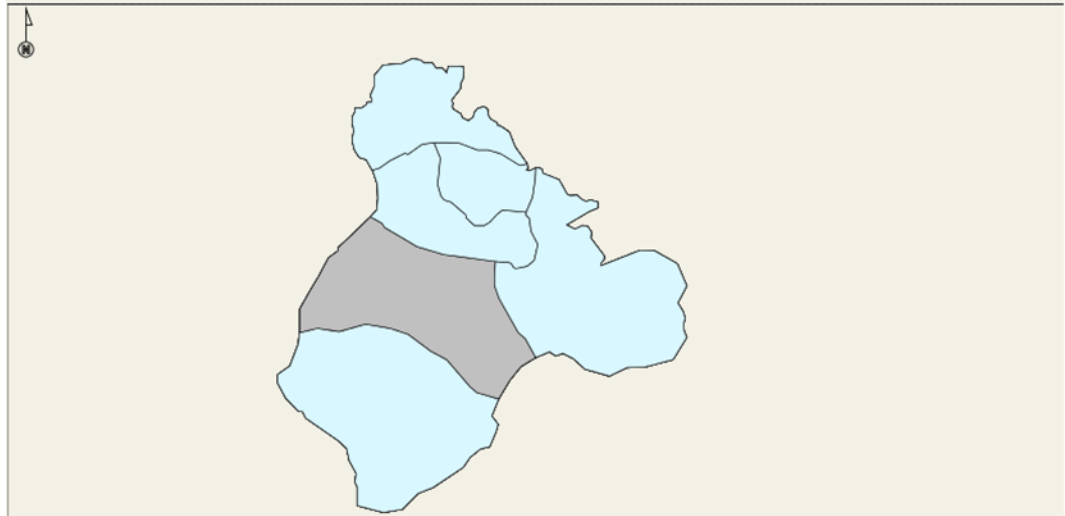


Figure 12.2.16 Ardahan GML file visualization

14 Conclusion

To sum up, in the research, the fundamental information has been given related with visualization of GML (Geographic Markup Language) data with XSLT(Extensible Style Sheet Transformations) in web based GIS(Geographic Information System). For this reason, all existing technologies have been defined, and explained. In Section 1, GIS geographic information systems and web-based geographic information systems have been introduced. In the second section, XML technologies have been introduced for understanding of the structure of the GML. In Section 3, it has been mentioned that GML which is preferred data format in web-based geographic information systems. In the fourth section, it has been provided to XSLT (Extensible Style Sheet Transformations) which is unique preferred conversion technology in web-based systems. In Section 5, SVG (Scalable Vector Graphic) technology which uses to produce high-quality graphic image files has been mentioned. In Section 6, it has been mentioned the topic of CSS styling mechanism which styles SVG vector graphics and GML files. In the seventh section, WFS (Web Feature Server) open services technology has been handled as the topic. This technology is necessary to obtain GML data from a GIS service provider. There are advantages of the approach. First of all, maps which are displayed as SVG format can be stored and saved on user's computer. In addition, it is possible to make change on the SVG objects and their properties by the user side. Secondly, thanks to the Web browsers, it is not necessary any extra imaging program for visualization of GML data. Another advantage is that it is possible to store spatial and non-spatial information which is related to spatial information in the same file. It is possible to two-dimensionally display thanks to the use of the XSLT transformation technology. At the same time, there is an advantage is that GML files can be displayed in different formats according to the documentation of the different conversion.

REFERENCES

- Ahmed Shaig**, 2001, “An Overview of Web based Geographic Information Systems”, New Zealand.
- Aras, İbrahim and Yildiz, Ferruh, January 2011**, “İnternet Tabanlı CBS'nin Sivil ve Askerî Amaçlı Acil Durum Uygulamalarında Kullanılmasında Yeni Bir Yaklaşım”, Ankara.
- Alison Meynert**, 2003, “Publishing GML data as interactive SVG maps”
url:”<http://www.svgopen.org/2003/papers/cleopatra/>”.
- Aydınöğlü A.Ç.**, 2003. “İnternet Tabanlı CBS Uygulaması: Trabzon İli Örneği. TMMOB Harita ve Kadastro Mühendisleri Odası 9. Türkiye Harita Bilimsel ve Teknik Kurultayı”, Ankara.
- Clark, James** 19 November 1999, XSL Transformations (XSLT) Version 1.0. W3C Recommendation. Copyright 1999 World Wide Web Consortium. URI: <http://www.w3c.org/TR/xslt/>.
- Çelikkilek, İbrahim**, 2011, “Her Yönüyle HTML 5”, Kodlab Yayıncılık
- Dempsey, Caitlin, 10 October 2001**, “Geography Markup Language (GML) 2.0 – Enabling the Geospatial Web”, <http://www.gislounge.com/geography-markup-language-gml-20-enabling-the-geo-spatial-web/>, (Access date: June 2014)
- Fallman, Sandra**, April 7, 2004, “Using WFS to build GIS support”, Umeå University, Sweden
- Fredrik and Klausen**, 2006, “GML processing with XSLT and spatial extensions”
url:”<http://www.svgopen.org/2003/papers/SvgExplorerOfGmlData/>”.
- JIANG Jun, YANG Chong-jun, REN Ying-chao, JIANG Miao**, 2008, “The Research and Application of WFS Based GML”
- Kenneth S.Herdy, David S.Burggraf and Robert D. Cameron**,
“http://www.svgopen.com/2008/papers/74-HighPerformance_GML_to_Transform_for_the_Visual_Presentation_of_Geographic_Data_in_WebBased_Mapping_Systems/#LuSantosSripadaKou07”, (Access date: June 2014)
- Kilinç, Deniz ve Prof Dr. Alp Kut**, “XSL Dönüşümleri ve Biçimleme”, Dokuz Eylül Üniversitesi, İZMİR

REFERENCES (continued)

- Komesli, Murat and Ünalır, Murat Osman**, 2004, Coğrafi Bilgi Sistemlerinde Coğrafi Verilerin GML (Geographic Markup Language) ile Modellenmesi, Bornova/İzmir.
- Liang ZOU**, 2004, "Modeling GML and SVG Data for Web GIS". Peking University, China.
- L.Lehto and T. Sarjakoski**, 2004, "Schema Translations by XSLT for GML encoded Geospatial Data in Heterogenous Web-Service Environment".
- Luca Piazza Bonati, Luciano Fortunati, Giuseppe Fresta**, 2003, "SVG Explorer of GML Data", link: "<http://www.svgopen.org/2003/papers/SvgExplorerOfGmlData>".
- OGC**, Geography Markup Language (GML) Standarts <http://www.opengeospatial.org/standarts/gml> (Access date: Haziran 2014)
- Pitney Bowes Software** , 2008, "MapXtreme Developer Guide v.7.1"
- Saxonica Extented Functions**,
<http://www.saxonica.com/documentation/extensibility/integratedfunctions/ext-fns-N.html> (Access date: june 2014)
- Sarı, Fatih; Erdi, Ali; Kirtiloğlu, Osman Sami**, "İnternet Tabanlı Coğrafi Bilgi Sistemi Uygulamalarında Geoserver Arcgis Server ve Google Map Api Entegrasyonu", Konya.
- Scalable Vector Graphics (SVG) 1.1 Specification**, 14 January 2003, W3C Recommendation, World Wide Web Consortium. URL: <http://www.w3.org/TR/2003/REC-SVG11-20030114/>,
- Taladoire, Gilles**, 2001, "Geospatial Data Integration and Visualization Using Open Standard".
- Tennakoon, February-2003**, "Visualization of GML data using XSLT", ITC, Netherlands,

REFERENCES (continued)

Wikipedia, 14 August 2010, “Coğrafi Bilgi Standartları”
“http://tr.wikipedia.org/wiki/Co%C4%9Frafî_bilgi_standartlar%C4%B1” (Access date: June 2014)

W3C XSLT, <http://www.w3.org/TR/xslt> (Access date: June 2014)